FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Data Mining para estudos de interacção entre fármacos e previsão de Efeitos Adversos de Fármacos

Marta Lopes

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Rui Camacho (FEUP)

Coorientador: Vítor Santos Costa (FCUP)

28 de Julho de 2018

Data Mining para estudos de interacção entre fármacos e previsão de Efeitos Adversos de Fármacos

Marta Lopes

Mestrado Integrado em Engenharia Informática e Computação

28 de Julho de 2018

Resumo

Hoje em dia existem medicamentos para um grande número de doenças e outros pequenos problemas. O uso irracional de medicamentos está a tornar-se um problema cada vez mais comum em todo o mundo. Os pacientes começam a tomar mais do que um medicamento de cada vez sem supervisão ou sem estarem devidamente informados e isso pode ter consequências nefastas. Além disso, alguns pacientes precisam de tomar vários medicamentos prescritos em simultâneo, como por exemplo pacientes idosos ou pacientes com doenças mais graves ou crónicas, e apesar de serem supervisionados é possível que alguns efeitos adversos possam surgir. Algumas interações que acontecem podem não ser conhecidas na altura da toma dos medicamentos.

A indústria farmacêutica está sempre interessada na redução dos custos e do tempo de produção de novos medicamentos. Mais ainda, os testes finais de um medicamento são feitos em pacientes reais, mas a amostra é sempre muito pequena e nem sempre é representativa dos consumidores finais do medicamento. Por outro lado, a interação medicamento-medicamento é um problema complexo que precisa de ser investigado e apesar de muitos estudos e experiências já terem sido realizados ainda há muito mais a ser descoberto. A previsão e a divulgação destes efeitos negativos ajudará os profissionais de saúde a levá-los em conta antes de prescrever certos medicamentos e também iria servir para informar o público em geral.

Uma abordagem para tentar melhorar a situação será compreender as interações conhecidas entre os medicamentos e seus efeitos adversos, fazendo então uma tentativa de prever novas interações por similaridade entre princípios ativos e explicá-los a partir da presença de efeitos adversos de um certo medicamento num certo paciente.

A análise de *Data Mining* irá ser útil para esta finalidade ao descobrir padrões em grandes quantidades de dados, revelando novas interações e explicando as antigas. Usando todos os dados disponíveis sobre os efeitos adversos de medicamentos e as suas interações com outros medicamentos pode ser possível chegar a um modelo capaz de prever efeitos adversos de medicamentos. A primeira coisa a fazer será verificar as interações conhecidas que provocam efeitos adversos entre certos medicamentos. Seguidamente irão ser utilizados algoritmos de previsão que utilizam as interações entre dois medicamentos, sendo estes representados pelos seus descritores moleculares presentes no princípio ativo a fim de perceber se é possível prever se a interação em cada par de medicamentos será causador de um efeito adverso. Finalmente os modelos de previsão criados pelos algoritmos irão ser avaliados tendo em conta métricas quantitativas, nomeadamente a *Accuracy* de cada modelo tendo em conta os pré-processamentos utilizados e os parâmetros modificados em cada algoritmo.

Com esta análise de previsão a partir do *Data Mining* pretende-se atenuar as falhas dos ensaios clínicos que são mais demorados e dispendiosos e nem sempre utilizam amostras representativas da população afetada.

Abstract

Nowadays there are drugs for a very large number of diseases and small aches. Irrational use of drugs is becoming an increasingly common problem in the world. Patients start taking more than one medicine at a time without supervision or without being properly informed and this could have many consequences. Apart from this, some patients need to take various prescribed drugs at a time, in particularly elderly patients, and despite being supervised there are still some less known or unknown adverse effects that can arise.

Drug industry is always concerned in reducing the production time of a new drug and its costs. The final tests for a drug are in actual patients but the sample is always quite small and sometimes not representative of the final consumers of the drug.On the other hand, the drug-drug interaction is a complex problem than needs to be investigated, and although many experiments and studies have been made there's still a lot more to be known. The prediction and disclosure of these negative effects would help health professionals to take them into account before subscribing certain drugs or even just to inform the general public.

One approach to improve the situation is to try to understand the known interactions between drugs and their side-effects and make an attempt at predicting new interactions by similarity between active principles and explaining them on account of the presence of adverse drug effects in patients.

Data Mining analysis may be helpful for that purpose, discovering patterns from large amounts of data, unveiling new interactions and explaining old ones. Using all the data available on drug effects and their interactions with another drugs it can be possible to achieve a model capable of predicting adverse drug reactions. The first thing to do is check the known interactions that cause adverse effects between certain medications. Then the prediction algorithms will use the interactions between two drugs, which are represented by their molecular descriptors present in the active ingredient in order to understand whether it is possible to predict whether the interaction in each pair of medicines will cause an adverse effect. Finally the prediction models created by the algorithms will be evaluated taking into account quantitative metrics, including the Accuracy of each model taking into account the pre processing used and the parameters modified in each algorithm.

Data Mining predictive studies may attenuate the shortcomings of clinical trials that are more time consuming and costly and not always use samples representative of the affected population.

Agradecimentos

Em primeiro lugar, gostaria de agradecer à Faculdade de Engenharia da Universidade do Porto que me acolheu durante 5 anos e me ajudou a crescer, a aprender e a conhecer todas as pessoas fantásticas que hoje fazem parte da minha vida. Quero também agradecer ao meu orientador Rui Camacho e ao meu co-orientador Vítor Costa que me acompanharam nesta etapa que hoje se finaliza sempre com vontade de me ensinar e apoiar e assim me ajudaram a alcançar o meu objetivo.

Um agradecimento especial aos colegas do meu ano (2013) de Engenharia Informática e Computação que também me apoiaram na realização desta dissertação e estiveram sempre disponíveis para ajudar no que fosse preciso. Agradeço também à Stefania Moscato que esteve sempre do meu lado e me deu o apoio e a força necessários, mesmo nos momentos em que a conclusão deste projeto parecia estar muito distante.

Por último, quero agradecer ao projeto *NanoSTIMA: Macro-to-Nano Human Sensing: Towards Integrated Multimodal Health Monitoring and Analytics/NORTE-01-0145-FEDER-000016* financiado pelo Programa Operacional Regional do Norte (NORTE 2020), sob o Acordo de Parceria PORTUGAL 2020, e através do Fundo Europeu de Desenvolvimento Regional (*European Regional Development Fund* - ERDF) pela disponibilização de dados utilizados para a realização deste projeto.

Marta Lopes

"Wanting to be someone else
is a waste of the person you are."

Kurt Donald Cobain

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

ADE	Adverse Drug Effect
ADReCS	Adverse Drug Reaction Classification
AID	Assay ID
CID	Compound ID
ChEBI	Chemical Entities of Biological Interest
CRISP-DM	CRoss-Industry Standard Process for Data Mining
CSD	Cegedim Strategic Data
DM	Data Mining
EAM	Efeito Adverso do Medicamento
FAERS	FDA Adverse Event Reporting System
FDA	Food and Drug Administration
GM	Graph Mining
ICD	International Statistical Classification of Diseases and Related Health Problems
ILP	Inductive Logic Programing
InchI	International Chemical Identifier
JSON	JavaScript Object Notation
KEGG	Kyoto Encycopledia of Genes and Genomes
MAUDE	Manufacturer and User Device Experience
MedDRA	Medical Dictionary for Regulatory Activities
ML	Machine Learning
NIH	US National Institutes of Health
RES	Recall Enterprise System
SDF	Structure Data Format
SID	Substance ID
SIDER	Side Effect Resource
SMILES	Simplified Molecular Input Line Entry Specification
SPL	Structured Product Labeling
STITCH	Search Tool for Interacting Chemicals
SVM	Support Vector Machine
THIN	The Health Improvement Network
Weka	Waikato Environment for Knowledge Analysis
XML	Extensible Markup Language

Capítulo 1

Introdução

O objetivo desta tese será mostrar a utilidade das técnicas de *Data Mining* (DM) para conseguir prever Efeitos Adversos de Medicamentos (EAM) que advém da interação entre medicamentos criando assim uma forma de previsão de EAMs a partir das interações entre medicamentos que seja mais precisa e menos dispendiosa que as técnicas mais utilizadas hoje em dia, que se baseiam em testes utilizando uma pequena amostra da população que muitas vezes acaba por não ser representativa do consumidor final, tornando mais difícil de obter resultados rigorosos.

Utilizando repositórios previamente escolhidos, onde um deles contém episódios reais que relatam a toma de medicamentos e os efeitos adversos associados, e a partir de técnicas de *Data Mining* será possível a realização de previsões com base em casos conhecidos e conhecimento da estrutura das moléculas associadas ao princípio ativo de cada medicamento.

As experiências que irão ser realizadas incluem uma avaliação quantitativa dos métodos utilizados para assim podermos prever os efeitos adversos da interação entre medicamentos. A avaliação quantitativa dos modelos gerados pelo DM dá-nos uma estimativa da qualidade das previsões.

1.1 Motivação e Objetivos

Efeitos Adversos de um Medicamento [1] (EAMs), são eventos indesejados que diferem daquele que seria considerado o resultado esperado de um tratamento. Estima-se que, nos Estados Unidos, os EAMs sejam responsáveis por cerca de 28% de todas as emergências hospitalares e 5% de mortes em hospitais. Como consequência, entre 30 a 150 biliões de dólares são gastos anualmente em hospitais para o tratamento destes efeitos adversos. Assim sendo, existe não só uma obrigação moral em encontrar tratamentos mais seguros como também um grande valor económico [PCCC15].

Atualmente, para saber mais sobre cada medicamento são feitos testes em pacientes sendo que o grupo de teste é apenas uma pequena amostra da população que muitas vezes acaba por ser uma amostra pouco precisa pois pode não demonstrar a verdadeira representação do consumidor final [MBL+04]. Podemos então concluir que a informação obtida sobre os efeitos adversos de

[1] Em Inglês, Adverse Drug Reactions (ADR) ou Adverse Drug Event (ADE)

medicamentos muitas vezes não é suficiente e é necessário tentar encontrar outras formas de chegar a resultados mais precisos.

Utilizando a quimioinformática e o *data mining* poderá ser possível utilizar as interações já conhecidas entre os medicamentos e os seus efeitos adversos para fazer uma tentativa de criar um modelo que torne possível prever novas interações por similaridade entre princípios ativos.

1.2 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 4 capítulos.

No Capítulo 2 é descrito o estado da arte onde são referenciados todos os repositórios, ferramentas e metodologias que são possíveis de usar no decorrer deste trabalho.

No Capítulo 3 toda a metodologia é especificada referindo também que repositórios, ferramentas e outras *frameworks* foram utilizadas para concretizar esta dissertação.

No Capítulo 4 são descritas todas as experiências realizadas bem como os seus resultados, incluindo também uma discussão sobre estes resultados.

No Capítulo 5 encontra-se uma conclusão sobre que trabalho foi realizado e que trabalho poderá ser realizado no futuro.

Capítulo 2

Fundamentos de Quimioinformática e Data Mining

2.1 Quimioinformática

A Quimioinformática é a aplicação de métodos de *computer science* para resolver problemas químicos. Estes métodos incluem técnicas de armazenamento, processamento e manipulação de dados químicos. Este campo foca-se principalmente em pequenas moléculas, e uma das maiores aplicações é encontrar novas estruturas que poderão potencialmente ser medicamentos.

Um medicamento ou fármaco, é uma substância química usada para tratar, curar, prevenir ou diagnosticar uma doença de forma a melhorar a vida de um ser-vivo. Os medicamentos são normalmente agrupados em classes, onde cada classe agrega um conjunto de medicamentos que têm estruturas químicas semelhantes, e são usadas para fins semelhantes. Cada medicamento tem um princípio ativo, ou seja, a substância que está biologicamente ativa (a que exerce efeito farmacológico).

Um *pathway* é uma série de interações entre as moléculas numa célula que a levam a sofrer certas mudanças. Os *pathways* dos fármacos contêm informações importantes que permitem compreender os mecanismos das acções de cada fármaco e o seu metabolismo bem como para o reposicionamento de cada medicamento. Estes *pathways* integram então um conjunto de reações químicas com um certo propósito e o objetivo dos medicamentos é alterar de forma benéfica a função anómala de um pathway [ZQC15]. Para ter uma melhor ideia de como as reações entre os *pathways* e os medicamentos se desenvolvem é necessário ter acesso a uma grande quantidade de informação, sendo que as bases de dados que contenham tanto *pathways* como quais as *pathways* em que cada medicamento atua são extremamente importantes.

Os compostos químicos podem ser representados por notações em linha (ex: SMILES, InChI) e existem vários algoritmos que podem calcular as estruturas 2D e 3D destes compostos. Para além disso existe um grande número de descritores moleculares.

Para que seja possível compreender e processar uma estrutura química através de um computador, é necessário que esta seja descrita numa sequência numérica única. Os descritores mole-

culares representam estruturas químicas incorporando vária informação. Um descritor molecular, que pode ser disposto numa matriz ou num vetor de *bits*, é o resultado final de um procedimento matemático e lógico que transforma informação química codificada numa representação simbólica de uma molécula [ABMA17].

Outra característica importante são os fragmentos estruturais (*fingerprints*) que podem ser calculadas usando as ferramentas que calculam os descritores ou com algoritmos de *Graph Mining* (GM).

2.1.1 Efeitos Adversos de Medicamentos

Adverse Drug Effects (ADE) ou, em português, Efeitos Adversos de Medicamentos (EAM) são definidos como reações não intencionais ou não desejadas aos fármacos para além dos efeitos terapêuticos antecipados aquando do seu uso clínico em doses normais [DKSL13].
Estes EAMs podem ser divididos em 5 tipos:

- **Tipo A**: Efeito Previsível;
- **Tipo B**: Efeito Imprevisível;
- **Tipo C**: Efeito crónico (contínuo);
- **Tipo D**: Efeito retardado;
- **Tipo E**: Efeito no fim do tratamento

Ainda assim nem sempre é possível encaixar um EAM numa destas categorias uma vez que a sua causa pode não ser conhecida. Os EAM são muitas vezes causados por interações não conhecidas entre fármacos.

2.1.2 Repositórios Web Relevantes

2.1.2.1 Medicamentos e Efeitos Adversos

openFDA

A openFDA[1] foi criada em Março de 2013 por Taha Kass-Hout enquanto Diretor-Chefe de Informação em Saúde na FDA (Food and Drug Administration). O objetivo deste projeto é criar um acesso fácil aos dados de acesso público para assim ser possível dar oportunidade ao público em geral para estar informado. Foram formatados, indexados e documentados dados públicos de alto valor, alta prioridade e escaláveis de forma a facilitar a utilização desses mesmos dados aos programadores e consumidores. Estes dados foram também disponibilizados através de um portal de acesso público que permite aos programadores usar de forma rápida e fácil no desenvolvimento de aplicações. Desde a criação do openFDA já foram desenvolvidas quatro APIs que disponibilizam acesso a dados sobre eventos adversos, rotulação de produtos farmacêuticos e relatórios que fornecem informação sobre o *recall* de qualquer produto regulado pelo FDA.

[1] https://open.fda.gov/

Já foram efetuados mais de 20 milhões de acessos à API, existindo mais de 6000 utilizadores registados na plataforma bem como 20000 endereços IP conectados e variadas aplicações desenvolvidas com a ajuda destas APIs.

Existem quatro fontes de informação principais disponíveis no openFDA [KHXM+16]:

- FAERS (FDA *Adverse Event Reporting System*) para medicamentos e produtos biológicos selecionados;
- SPL (*Structured Product Labeling*) também para medicamentos e produtos biológicos selecionados;
- RES (*Recall Enterprise System*) para avisos de *recall* e também levantamentos do mercado e alertas de segurança, para medicamentos, produtos biológicos selecionados, dispositivos e alimentos;
- MAUDE (*Manufacturer and User Device Experience*) para relatórios de efeitos adversos.

PubChem

Criado em 2004 com o objetivo de ser um componente da *Molecular Libraries Roadmap Initiatives of the US National Institutes of Health (NIH)*, o PubChem[2] é, hoje em dia, um repositório público que contém informação sobre substâncias químicas e as suas atividades biológicas. Com o seu rápido crescimento, este repositório tornou-se num recurso-chave sobre informação química, utilizado por comunidades científicas das mais variadas áreas como quimioinformática, biologia química, medicina química e pesquisa de medicamentos [KTB+16].

A informação contida no PubChem é produzida por mais de 350 contribuidores[3], incluindo laboratórios de várias universidades, agências governamentais, empresas farmacêuticas, entre outros. Em Setembro de 2015, este repositório dispunha de mais de 157 milhões descrições de substâncias químicas fornecidas pelos depositantes, 60 milhões de estruturas químicas únicas e 1 milhão de descrições de ensaios biológicos. Toda esta informação é organizada em três bases de dados interligadas:

- **Substance**[4]: contém descrição de substâncias e mais de 254 milhões de SIDs (*Substance ID*)
- **Compound**[5]: contém estruturas químicas únicas que são extraídas da base de dados de substâncias (mais de 60 milhões de CIDs (*Compound ID*)
- **BioAssay**[6]: contém descrições de ensaios biológicos em substâncias químicas. (mais de 1 milhão de AIDs (*Assay ID*))

[2] https://pubchem.ncbi.nlm.nih.gov/
[3] https://pubchem.ncbi.nlm.nih.gov/sources/
[4] https://www.ncbi.nlm.nih.gov/pcsubstance
[5] https://www.ncbi.nlm.nih.gov/pccompound
[6] https://www.ncbi.nlm.nih.gov/pcassay

O PubChem fornece também várias vias de acesso aos dados através da programação, incluindo o PUG-REST[7], um ponto de acesso robusto e o mais simples de utilizar. A informação necessária para fazer um pedido PUG-REST pode ser codificada apenas num URL e incorporado em páginas web ou processos de trabalho mais complexos fornecendo um acesso conveniente à informação contida nos relatórios do PubChem, acesso este que não é possível com outros serviços PUG [KTB+16].

RxNav - Drug Interaction RESTful API

O RxNav é um navegador para várias fontes de informação sobre fármacos, como o RxNorm[8], RxTerms[9] e MED-RT (*Medication Reference Terminology*) [BPZM13]. Este navegador encontra fármacos no RxNorm a partir de nomes e códigos que constam no seu vocabulário. O RxNav apresenta ligações entre fármacos, de marca ou genéricos, os seus princípios ativos, os seus componentes e outras marcas relacionadas. Os arquivos do RxTerms sobre um determinado fármaco também podem ser acedidos a partir do RxNav, bem como informação clínica proveniente do MED-RT.

A API RESTful para Interações entre Fármacos é um *web service* desenvolvido na Biblioteca Nacional de Medicina para aceder a informações sobre interações entre fármacos. Esta API utiliza o REST que é um estilo de arquitetura de *software* para sistemas distribuídos. Este estilo de arquitetura consiste em ter um cliente que inicia os pedidos aos servidores, estes vão então processar os pedidos e responder adequadamente.

Este *web service* é implementado utilizando HTTP e pode ser considerado como sendo uma coleção de recursos, especificados com URIs. Algumas características desta API são:

- O URI base para aceder ao *web service* é *https://rxnav.nlm.nih.gov/REST/interaction*
- O esquema do ficheiro que descreve o formato XML que pode ser acedido aqui[10]
- O *web service* pode retornar informação nos formatos XML ou JSON. O formato pode ser especificado ao acrescentar a extensão (.xml ou .json) (Exemplo de como obter uma resposta JSON: *https://rxnav.nlm.nih.gov/REST/interaction/interaction.json?rxcui=341248*)
- O *web service* apenas suporta o método HTTP GET, dado que apenas é possível obter informação

[7] https://pubchem.ncbi.nlm.nih.gov/pug_rest/PUG_REST.html
[8] https://www.nlm.nih.gov/research/umls/rxnorm/
[9] https://wwwcf.nlm.nih.gov/umlslicense/rxtermApp/rxTerm.cfm
[10] https://rxnav.nlm.nih.gov/interactionrest.xsd

ADReCS

A ***A**dverse **D**rug **R**eaction **C**lassification **S**ystem* é uma base de dados em XML que é mantida por investigadores da Universidade de Xiamen [CXP+14]. Integra várias base de dados de repositórios médicos, tais como: Daily Med[11], MedDRA[12], SIDER[13], DrugBank[14], PubChem[15], UM LS,[16] entre outros. Todos estes dados provêm de várias fontes: registos por parte de consumidores, resultados laboratoriais, registos médicos e registos farmacêuticos. Como todos estes dados têm diferentes origens, os nomes utilizados para cada medicamento ou para cada efeito adverso podem variar e para prevenir isso o ADReCS padroniza os dados utilizando a MedDRA e o UM LS como principais referências. Para além disso é também gerada uma classificação hierárquica de EAMs de quatro níveis para facilitar a pesquisa. A cada EAM é atribuído um ID único com quatro campos separados por '.' (exemplo: xx.xx.xx.xxx).

Neste momento, a ADReCS disponibiliza informação de 1698 medicamentos e 7668 efeitos adversos, num total de 157 246 ocorrências medicamento - efeito adverso, 50 717 ocorrências efeito adverso-gene, 2692 ocorrências efeito adverso-proteína.

ICD

A *International Statistical Classification of Diseases and Related Health Problems* (ICD), mantida pela *World Health Organization* é um sistema de classificação das doenças que proporciona um sistema de códigos de diagnóstico para classificar doenças, que podem incluir sinónimos nas classificações de sinais, sintomas, queixas, circunstâncias sociais e causas externas de doença ou lesão [Wor11].

Este sistema é desenhado de forma a mapear as condições de saúde organizando-as por categorias mais gerais com variações mais específicas, associando para cada um, um código de seis caracteres. Neste momento, a versão mais recente é a ICD-10 apesar de que a versão mais utilizada na prática é a ICD-9.

THIN

The Health Improvement Network (THIN) contém, atualmente, registos médicos de cerca de 11.1 milhões de pacientes (3.7 milhões pacientes ativos) do Reino Unido [MS09]. Todos os dados são anónimos, processados e validados pela *CSD Medical Research UK*.

Nesta base de dados é mantida informação sobre os pacientes, diagnósticos feitos e prescrições dadas. São disponibilizados alguns detalhes sobre o paciente (se é fumador, altura, peso, imunidades, gravidez, entre outros) que podem influenciar algumas considerações em relação aos efeitos adversos que surjam.

[11] https://dailymed.nlm.nih.gov/dailymed/
[12] https://www.medra.org/
[13] https://github.com/mkuhn/sider
[14] https://www.drugbank.ca/
[15] https://pubchem.ncbi.nlm.nih.gov/
[16] www.stackexchange.com

MedDRA

O *Medical Dictionary for Regulatory Activities* (MedDRA) é uma terminologia médica padronizada, internacionalizada e validada clinicamente que é utilizada pelas autoridades reguladoras na indústria farmacêutica de forma a facilitar a partilha de informação na comunidade ao criar uma terminologia uniformizada [Int13].

2.1.2.2 Repositórios de Biologia e Interações moleculares

ChEBI

Chemical Entities of Biological Interest[17] é um dicionário gratuito de entidades moleculares focado em compostos químicos "pequenos". As entidades moleculares são produtos naturais ou sintéticos que podem intervir nos processos de organismos vivos. Para além disso o ChEBI contém também uma classificação ontológica onde as relações entre entidades moleculares, ou classes de entidades, e os seus pais e/ou filhos são especificados [DDmE+08].

Para ser possível criar o ChEBI foi necessário incorporar e juntar informação de várias fontes. Para o lançamento inicial desta plataforma foram maioritariamente utilizadas as seguintes fontes de informação:

- **IntEnz** (*Integrated relational Enzyme database*)
- **KEGG COMPOUND**
- **Chemical Ontology**

A base de dados do ChEBI é relacional e é implementada num *server Oracle* fornecendo várias informações sobre uma determinada substância, tais como:

- **ChEBI ID**: um identificador único e estável para a entidade
- **Nomes ChEBI**
- **Definição**
- **Diagramas Estruturais**
- **IUPAC InChI**
- **SMILES**
- **Fórmula**
- **Ontologia**: a Ontologia do ChEBI consiste nas seguintes sub-ontologias:
  - **Estrutura Molecular** - classifica uma entidade molecular ou partes dela de acordo com a sua estrutura;

[17] ttp://www.ebi.ac.uk/chebi/

- **Papel Químico** - classifica entidades com base no seu papel num contexto químico.
- **Papel Biológico** - classifica entidades com base no seu papel num contexto biológico (exemplos: antibiótico, hormona);
- **Aplicação** - classifica entidades, quando aplicável, com base no uso dado pelos humanos (exemplos: pesticida, medicamento, combustível);
- **Partícula Subatómica** - classifica partículas mais pequenas que átomos.

DrugBank

A DrugBank[18] é uma base de dados online, gratuita e bastante abrangente. É um recurso para áreas da bioinformática e da quimioinformática, combinando dados detalhados sobre cada medicamento e outras informações como a sua descrição, estrutura e *pathway* em que atua. Por ter uma grande esfera de ação com referências variadas e descrições excecionalmente detalhadas, esta base de dados é mais tratada como uma enciclopédia [WFG+17].

É possível pesquisar por medicamentos, categoria, gene, reação, *pathway*, classe, proteína alvo ou indicações de uso.

Podem ser encontradas nesta base de dados cerca de 10 950 entradas entre as quais 5086 são experimentais sendo que DrugBank agrupa e revê informação em mais de 50 bases de dados/aplicações Web.

SIDER

O *Side Effect Resource*[19] (SIDER) é uma base de dados com informação assimilada a partir da variada informação pública sobre efeitos adversos de medicamentos validada com recurso à MedDRA, STITCH e PubChem [KLJB15]. A informação disponível neste recurso inclui: frequência de cada efeito adverso, classificação do medicamento e do efeito adverso e também ligações para outras fontes de informação.

O SIDER contém, atualmente, informação sobre 5868 efeitos adversos de 1430 medicamentos entre 139 756 pares medicamento-efeito adverso.

STITCH

A *Search Tool for Interacting Chemicals*[20] (STITCH) é uma ferramenta que integra várias fontes de informação sobre 430 000 proteínas num único recurso [SSvM+15]. Para além de ser uma base de dados bastante abrangente permite também ao utilizador ter uma visão global dos efeitos adversos das proteínas em cada interação. Cada interação prevista tem uma pontuação do nível de confiança.

Este servidor Web/base de dados para além de obter informação de diversas base de dados utiliza também uma ferramenta de *text mining* para importar interações proteína-proteína trazidas de textos científicos.

[18] https://www.drugbank.ca/
[19] http://sideeffects.embl.de/
[20] http://stitch.embl.de/

KEGG

A *Kyoto Encycopledia of Genes and Genomes*[21](KEGG) contém informação sobre *pathways* metabólicos a partir de uma variedade de espécies. É uma das bases de dados mais completas e utilizadas [KSK+16]. Atualmente, a KEGG tem mais de 15 000 componentes e 7742 medicamentos.

2.1.3 Ferramentas para Quimioinformática

PaDEL

O PaDEL-Descpritor[22] é um *software standalone*, desenvolvido em Java, que calcula descritores moleculares e *fingerprints* [Yap10]. Atualmente, esta ferramenta consegue calcular 1875 descritores e 12 tipos de *fingerprints*. Tanto os descritores moleculares como as *fingerprints* são calculadas com a ajuda do *The Chemistry Development Kit* [23]. Este *software* pode também ser usado como extensão do RapidMiner ou do KNIME.

Open Babel

O Open Babel[24] é um software gratuito e *open-source* maioritariamente utilizado para pesquisa, análise e conversão de variados dados químicos, suportando 111 formatos de arquivos químicos [OBJ+11]. Este software fornece também uma pesquisa por substrutura baseado no SMILES (*Simplified Molecular Input Line Entry Specification*) que é uma forma de representar estruturas químicas usando caracteres ASCII e o cálculo da *fingerprint* (impressão digital) de uma molécula, obtida a partir do mapeamento de todas as subestruturas lineares e de anel de uma molécula. Através da comparação entre impressões digitais facilitam a identificação de moléculas semelhantes, reduzindo também o tempo de pesquisa de uma molécula. O Open Babel possibilita também a criação de coordenadas 2D e 3D de uma molécula obtidas a partir do seu identificador SMILES e a conversão entre estas três estruturas.

2.2 Data Mining

Data Mining é o processo de descobrir padrões e adquirir conhecimento a partir de grandes quantidades de informação. A informação pode ser obtida a partir de bases de dados, armazéns de dados, outros repositórios de informação ou dados que são inseridos no sistema dinamicamente [HKP12].

A metodologia habitualmente utilizada em DM inclui uma sequência de passos que culmina com a extração de conhecimento de dados [HKP12]:

[21] https://www.genome.jp/kegg/
[22] http://www.yapcwsoft.com/dd/padeldescriptor/
[23] https://sourceforge.net/projects/cdk/
[24] http://openbabel.org/wiki/Main_Page

1. **Limpeza dos dados** - extração de dados incorretos e/ou irrelevantes
2. **Integração dos dados** - combinação de dados de diferentes fontes
3. **Seleção de dados** - extração de todos os dados relevantes
4. **Transformação dos dados** - realização de operações de síntese ou agregação de dados
5. **Data Mining** - processo em que diferentes métodos podem ser aplicados com o objetivo de extrair padrões de dados construindo assim modelos para os dados
6. **Avaliação dos modelos** - identificação dos padrões que realmente representam conhecimento
7. **Utilização do conhecimento** - utilização do conhecimento adquirido para a aplicação no uso pretendido

2.2.1 Metodologia CRISP-DM

O *CRoss-Industry Standard Process for Data Mining* (CRISP-DM) define um projeto de maneira cíclica onde várias iterações podem ser utilizadas para permitir que o resultado final esteja de acordo com o objetivo pretendido [ML11]. Esta metodologia tem então seis fases [CRI08]:

1. *Business understanding* - Esta fase inicial serve para entender os objetivos do projeto e os requisitos de uma perspetiva de negócio para depois converter todo o conhecimento numa definição de um problema de *data mining* e num plano preliminar para chegar ao objetivo final;
2. *Data understanding* - Nesta fase é feita a recolha e análise de dados;
3. *Data preparation* - Esta fase serve para construir o *dataset* final a partir dos dados recolhidos na fase anterior;
4. *Modeling* - Nesta fase é criado um modelo que representa o conhecimento adquirido a partir dos *datasets*;
5. *Evaluation* - Neste ponto o modelo, ou modelo obtido irá ser avaliado e todos os passos executados para a construção do modelo são revistos de forma a confirmar que vão de encontro aos objetivos de negócio. Caso o modelo obtido não seja suficiente, irá ser feita uma nova iteração do CRISP-DM, se o modelo estiver de acordo com o esperado então passamos à fase seguinte;
6. *Deployment* - Esta fase final servirá para organizar e apresentar todo o conhecimento adquirido de forma a que possa ser utilizado.

É possível ainda observar a representação visual destas fases na Figura 2.1 onde se pode constatar que este processo nem sempre é linear pois existe a possibilidade de serem feitos retrocessos em algumas das fases como nas fases de *Data Understanding*, *Modeling* ou *Evaluation*.

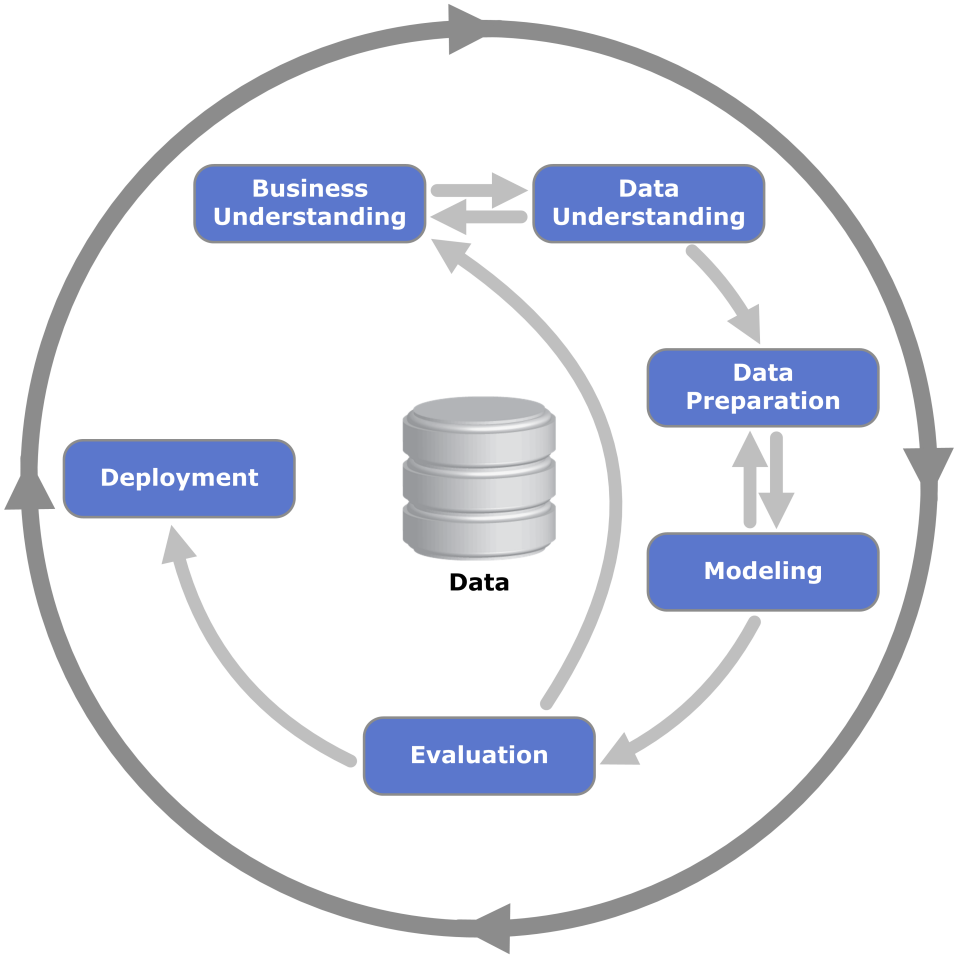

Figura 2.1: Fases da Metodologia CRISP-DM - criado por *Kenneth Jensen* baseado em [IBM11]

2.2.2 Tarefas de Data Mining

O Data Mining tem como principais tarefas as de Previsão e de Descrição. As tarefas de Previsão criam um modelo a partir da informação disponível que irá ser útil na previsão de valores desconhecidos ou os futuros valores de um certo *dataset*. A Previsão pode ser vista como a construção e uso de um modelo de forma a poder avaliar objetos não classificados. Posto isto, a classificação e a regressão acabam por ser dois grandes tipos de técnicas de previsão onde a classificação é usada para prever valores discretos nominais e a regressão prevê valores contínuos.

As tarefas de Descrição, na sua generalidade, encontram modelos que irão descrever os padrões acrescentando também informação nova e importante a partir de um *dataset*.

Existem inúmeras tarefas, sejam de previsão ou de descrição, sendo apresentadas em baixo as que podemos considerar principais.

Classificação

A Classificação é o processo de encontrar um grupo de modelos ou funções que irão descrever ou distinguir um conjunto de classes ou conceitos num *dataset* para permitir o uso destes modelos na previsão das classes ou conceitos em falta num determinado objeto. O modelo deriva de uma análise ao conjunto de dados chamado de *training data* onde todas as classes dos objetos são conhecidas.

Podemos descrever este processo mais pormenorizadamente em dois passos. Primeiramente um modelo é construído que irá descrever as classes ou conceitos de dados. O modelo é então construído baseado na análise dos tuplos descritos pelos atributos.

Cada tuplo irá pertencer a uma classe, determinada a partir de um dos atributos, a *class label attribute*. No contexto da Classificação, estes tuplos são tratados como amostras, exemplos ou objetos. Os tuplos de dados são analisados para construir um modelo, a partir do conjunto de dados de treino. Os tuplos individuais que irão fazer o conjunto de dados de treino são as amostras de treino que são escolhidas aleatoriamente a partir da população da amostra. Dado os atributos de cada amostra de treino são fornecidos, este passo é chamado de aprendizagem supervisionada (o modelo de aprendizagem é supervisionado, ou seja, é dito a que classe cada amostra de treino pertence) [HKP12].

A Regressão e a Classificação acabam por ser técnicas de *data mining* para resolver problemas semelhantes. Enquanto que a classificação atribui dados em categorias discretas, a regressão é usada para prever um valor numérico ou contínuo.

Clustering

Ao contrário da classificação, o *clustering* analisa dados de objetos sem os ter associados a qualquer classe. As *labels* de cada classe não estão presentes no conjunto de dados de treino dado que elas simplesmente não existe. Os objetos são agrupados (*clustered*) usando como critério o principio da maximização da similaridade intra-classes e minimização inter-classes. Isto é, os grupos de objetos são formados para que dentro de cada grupo (ou *cluster*) exista uma grande semelhança entre os objetos, mas que quando comparados objetos de grupos diferentes, estes sejam bastante diferentes. Cada grupo que é formado é visto como uma classe de objetos, de onde podem ser derivadas regras. O *clustering* pode também facilitar a formação de taxonomias, ou seja, a organização de observações numa hierarquia de classes que agrupa eventos semelhantes [HKP12].

2.2.3 Ferramentas de Data Mining

Weka

O *Weka* é uma ferramenta open-source e desenvolvida em Java. Disponibiliza um grande conjunto de algoritmos *data mining* e ferramentas de pré-processamento de dados. Alguns dos algoritmos que podemos encontrar nesta ferramenta são: regressão, classificação, *clustering* e regras de associação entre outros [WFHP16].

Esta ferramenta dispõe de diversos algoritmos úteis para o pré-processamento de dados.

Para além disso, o Weka permite carregar dados a partir de ficheiros, urls ou bases de dados e suporta vários formatos como ARFF, CSV, LIBSVM e C4.5. É também possível adicionar *plugins* como o Bioweka que é utilizado em áreas como biologia, bio-informática e bioquímica [GSZ07]. É disponibilizada também a possibilidade de exportar ou reutilizar modelos. No entanto a visualização dos modelos finais nesta ferramenta não é de grande qualidade.

O *Weka* possui também filtros como o *Randomize* que troca a ordem das instâncias de forma aleatória e o *RemovePercentage* que remove uma certa percentagem de um *dataset* sendo possível depois inverter a seleção e selecionar a percentagem restante. A partir destes dois filtros é possível obter vários *training* e *teste sets* a partir de um *dataset* inicial. Outro filtro também disponibilizado para o pré-processamento de *datasets* é o *Standardize*. Este filtro padroniza todos os atributos numéricos num determinado *dataset* para ter média zero e distância em % do desvio padrão. Este filtro não é aplicado no atributo de classe, se este estiver definido. É também disponibilizado o filtro *Remove* que serve para remover atributos de acordo com o seu índice, podendo ser removidos vários atributos em simultâneo.

No *Weka* existem também seletores de atributos, que vão selecionar e organizar os atributos de acordo com um conjunto de fatores. Um destes seletores é o *CorrelationAttributeEval* que avalia o valor de um atributo medindo a correlação (Pearson[25]) entre este atributo e a classe. Neste seletor se o resultado for 1 ou -1 significa que as duas variáveis têm uma correlação perfeita positiva ou negativa respetivamente. Se o valor for 0 significa que as variáveis não dependem linearmente uma da outra.

R

O *RStudio* é uma ferramenta *open source* desenvolvida em C++, para utilização de linguagem R [RSt15], que usa o *framework* Qt[26] para a sua GUI.

Este *software* está disponível em duas versões: *RStudio Desktop* para ser utilizado localmente e o *RStudio Server*, que permite o acesso ao *RStudio* a partir de um *browser*.

RapidMiner

Ferramenta *open source* desenvolvida em Java que suporta todas as etapas do processo de *data mining*. O *RapidMiner* [cit12] utiliza XML internamente para uniformizar os seus dados, dados estes que poderão ser extraídos de várias fontes: Excel, Access, Oracle, IBM DB2, Microsoft SQL Server, ficheiros de texto, entre outros.

Este programa executa todos os algoritmos que o *Weka* contém e ainda disponibiliza outros, diminuindo a necessidade de escrever código devido a todas as suas funcionalidades, sendo estas

[25] O coeficiente de correlação de Pearson mede o grau da correlação (e a direcção dessa correlação - positiva ou negativa) entre duas variáveis de escala métrica.

[26] https://www.qt.io/

bastante intuitivas. A visualização do resultado final é feita automaticamente e em vários formatos: gráficos de barras, densidade, 3D, etc.

KNIME

O *KNIME* é baseado na plataforma IDE, que poderá ser plataforma de desenvolvimento ou de *data mining* [BCD+07]. É desenvolvido também em Java e utiliza *plugins* para integrar funcionalidades adicionais. Sem estes *plugins* inclui algoritmos de integração, transformação e visualização de dados.

2.2.4 Data Mining Multi-relacional

O *Data Mining* convencional ou proposicional contém fortes limitações quando pretendemos representar os dados na expressividade da linguagem que codifica os modelos induzidos. Nos algoritmos proposicionais os dados são codificados no formato atributo/valor (equivalente a uma única tabela de uma base de dados relacional). É difícil representar neste formato dados com estrutura tais como mapas, estrutura de moléculas etc.

Os algoritmos deste tipo de *data mining*, como Árvores de Decisão, SVMs, ANN, K-NN, entre outros, podem ser encontrados nas ferramentas referidas anteriormente (*Weka, RapidMiner*, etc) onde os dados têm que ser descritos num formato atributo-valor e guardados numa única tabela de uma base de dados ou numa folha Excel.

Dada estas graves limitações do *Data Mining* proposicional, é recomendada a utilização de um *Data Mining* Multi-relacional, mais especificamente o Inductive Logic Programming (ILP), que não tem estes entraves a nível de representação de dados com estruturas nem a construir modelos sofisticados. O modelo multi-relacional pode então tratar sem problemas várias relações em simultâneo tendo em conta, por exemplo, dados de todas as tabelas de uma base de dados. Este modelo também não apresenta qualquer tipo de problema em codificar grafos o que permite ser utilizado de forma a codificar moléculas, o que irá ser necessário para este projeto.

2.2.5 Algoritmos de Classificação

Os algoritmos de classificação vão utilizar um conjunto pré-classificado de exemplos e construir um modelo capaz de classificar novos casos. É utilizada uma aprendizagem supervisionada onde as classes são conhecidas para os exemplos utilizados na construção do classificador. Um classificador poderá ser um conjunto de regras, uma árvore de decisão, uma rede neuronal, etc. Existem vários algoritmos de classificação sendo que nesta subsecção irão ser descritos alguns deles.

2.2.5.1 SVM

Support Vector Machine é um algoritmo de *machine learning* que pode ser usado para classificação ou regressão mas é principalmente usado para problemas de classificação [SC08]. Neste

algoritmo, é dado como entrada um conjunto de dados e, para cada entrada, é previsto qual das duas classes possíveis a entrada vai pertencer. Pode-se então dizer que o SVM é um classificador linear binário não probabilístico.

O modelo SVM representa os *data items* como pontos no espaço, que vão sendo mapeados de forma a que cada categoria seja dividida por um espaço claro e amplo. A partir de um conjunto de dados de treino em que cada um é marcado como pertencente a uma de duas categorias, o algoritmo de treino do SVM vai construir um modelo que vai atribuir uma das categorias a novos *data items*, e mapeando-os no lado do espaço pertencente à sua categoria.

Por outras palavras, o SVM encontra uma linha de separação (hiperplano) entre os dados de duas classes. Como é possível ver na Figura 2.2, existe uma linha que vai tentar maximizar a distância entre os pontos mais próximos em relação a cada uma das classes.

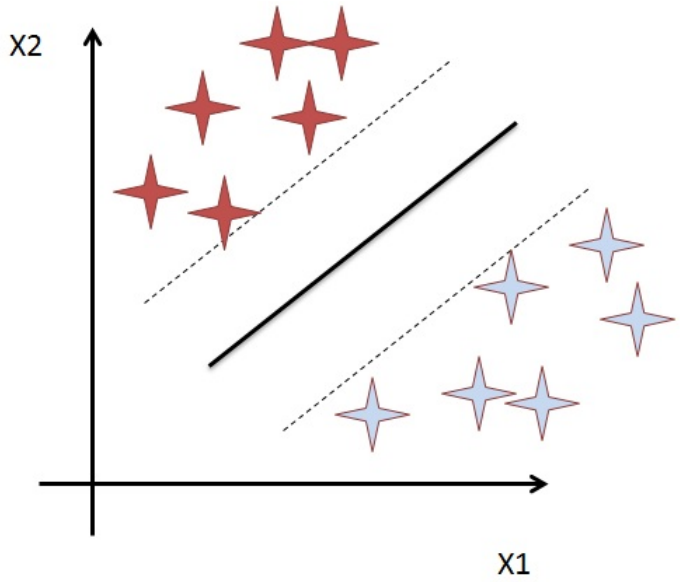

Figura 2.2: Separação dos dados no hiperplano - criado por *Roni Shouval*

Neste algoritmo um dos parâmetros importantes a avaliar é o *kernelType* (classe de algoritmos para análise de padrões) que pode ser variado entre:

- **linear**: u'*v
- **radial basis function**: $exp(-gamma * |u-v|^2)$;
- **polynomial**: $(gamma * u' * v + coef0)^{d}$[27]
- **sigmoid**: $tanh(gamma * u0 * v + coef0)$

[27] degree

2.2.5.2 Random Forest

O algoritmo *Random Forest* é um tipo de *ensemble learning*, onde são gerados vários classificadores do mesmo tipo (*Bagging*) e os seus resultados são combinados [Bre01].

Este algoritmo gera várias árvores de decisão diferentes (algoritmo CART) e combina o resultado da classificação de todas. Este algoritmo acaba por ser mais vantajoso que o algoritmo *Decision Tree* devido a esta combinação de modelos. Na Figura 2.3 podemos ver um exemplo de *Random Forest* com duas árvores de decisão.

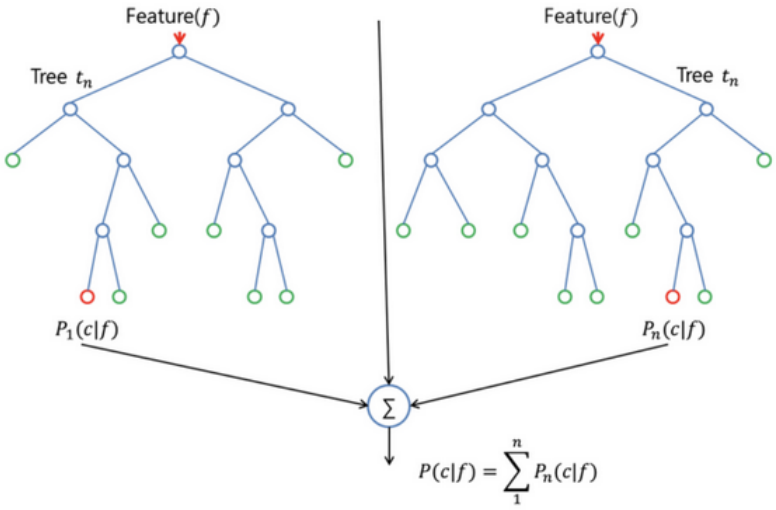

Figura 2.3: Algoritmo *Random Forest* - retirado de Analytics Vidhya

No algoritmo *Random Forest* podemos então ter em consideração vários parâmetros, sendo que um dos mais importantes é o *numIterations* que é o número de árvores geradas no decorrer do processamento deste algoritmo. Geralmente, quanto mais árvores utilizadas melhores vão ser os resultados. Apesar disso, a partir de um certo ponto, o beneficio do aumento da *performance* com a aprendizagem a partir de um grande número de árvores não irá compensar em relação ao custo do tempo de computação para aprender com as árvores adicionais.

2.2.5.3 k-NN

O algoritmo k-NN, k-*nearest* neighbours (em Português, k-vizinhos mais próximos) é um método usado para classificação ou regressão [Pet09]. É um tipo de *lazy learning* (aprendizagem "preguiçosa"), onde a função é aproximada localmente e toda a computação é atrasada até à classificação. Este algoritmo está entre de *Machine Learning* algoritmos mais simples.

Na classificação um objeto é classificado pela maioria de votos dos seus vizinhos, sendo que lhe é atribuída a classe mais comum entre os seus k vizinhos mais próximos (k é um inteiro positivo, normalmente pequeno). Se k=1 então o é atribuído ao objeto a classe do único vizinho mais próximo.

Na Figura 2.4 pode-se ver um exemplo da classificação k-NN. Poderá ser atribuída a classe de quadrados azuis ou a classe de triângulos vermelhos à amostra de teste (circulo central).

Se k = 3 (círculo de linha continua) é atribuída a classe triângulo porque existem 2 triângulos e apenas 1 quadrado dentro do círculo interior. Se k = 5 (círculo de linha tracejada) é atribuída a classe quadrado pois existem 3 quadrados e apenas 2 triângulos.

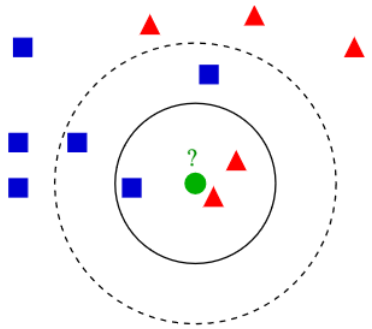

Figura 2.4: Algoritmo *k-NN* - Classificação - criado por *Antti Ajanki*

Neste algoritmo existem vários parâmetros que podem sofrer variação, tais como o *K* que é o número de vizinhos a utilizar e a *distanceFunction* (função de distância) que calcula distância métrica para quantificar a semelhança definindo assim os vizinhos e a noção de próximo em geral. Existem várias funções de distância disponíveis no Weka, tais como:

- *Euclidean distance*: a distância euclidiana é a distância entre dois pontos, que pode ser provada pela aplicação repetida do teorema de Pitágoras. Irá ser então o comprimento da linha que conecta os dois pontos.
- *Manhattan distance*: a distância de *Manhattan* é a distância entre dois pontos numa grelha baseando-se estritamente na soma dos caminhos horizontais e ou verticais (ou seja, ao longo das linhas da grelha).

A diferença entre a *Euclidean distance* e a *Manhattan distance* pode ser observada na Figura 2.5.

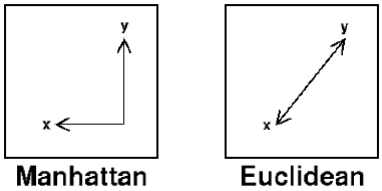

Figura 2.5: Comparação entre *Euclidean distance* e *Manhattan distance* - [MC14]

2.2.5.4 J48

O algoritmo J48 é uma implementação no Weka do algoritmo C4.5 de *Ross Quinlan* [Qui93]. Este algoritmo constrói uma árvore de decisão para cada conjunto de dados. O objetivo é uma

generalização progressiva de uma árvore de decisão até alcançar o equilíbrio entre flexibilidade e precisão. Na Figura 2.6 é possível ver um exemplo de um modelo (árvore de decisão) construído por este algoritmo.

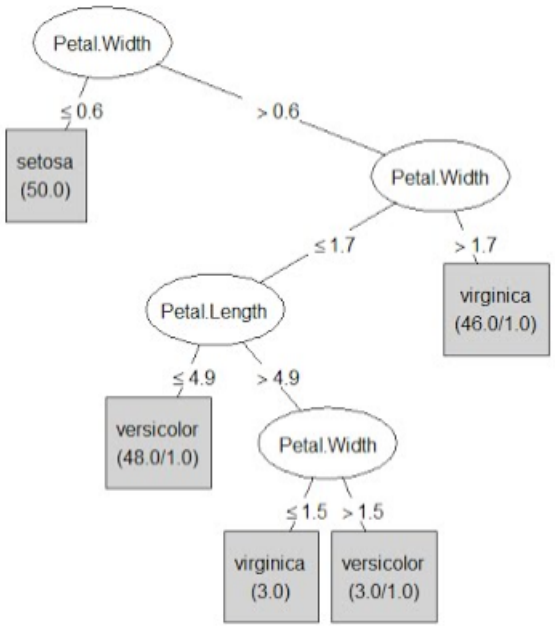

Figura 2.6: Exemplo de uma utilização do Algoritmo J48 - retirado de Business Intelligence UOC

Um dos parâmetros onde a sua variação pode ter alguma importância no desempenho do algoritmo é o *minNumObj* que é o número mínimo de instâncias por folha, este parâmetro vai garantir o necessário para continuar o processo de crescimento da árvore. Se, durante o processo de construção da árvore um nó tiver um número de exemplos igual ou inferior a *minNumObj* esse nó é transformado em folha.

2.2.5.5 AdaBoost

AdaBoost significa *Adaptive Boosting* (em Português, "impulso adaptativo"). *Boosting* é uma técnica onde vários algoritmos de aprendizagem simples são combinados para criar uma previsão de alta precisão [Sch13]. Foi a partir desta técnica, que Yoav Freund e Robert Schapire criaram este algoritmo juntando-lhe ainda três condições:

1. Os classificadores devem ser treinados em exemplos de treino suficientes;

2. Deve ser proporcionado um bom ajuste para estes exemplos, com um nível baixo de erro no treino;

3. Devem ser utilizados modelos simples dado que modelos simples são melhores que modelos demasiado complexos

No seu funcionamento, o *AdaBoost* chama então um classificador fraco iterativamente. Para cada chamada a distribuição de pesos vai sendo atualizada para indicando a importância do exemplo no conjunto de dados usado para classificação. Em cada iteração é verificado se os pesos para cada exemplo estão corretamente classificados ou se devem ser modificados e assim permite ao classificador trabalhar em mais exemplos.

Dois dos parâmetros onde é significativo estudar a sua variação são então o *numIterations* que é o número de iterações a serem efetuadas e o *classifier* que vai escolher o classificador base a ser utilizado.

2.2.5.6 Rede Neuronal Artifical

Uma Rede Neuronal Artificial é uma técnica de processamento de informação que se assemelha ao processamento de informação feito pelo cérebro [Has95]. Tem um grande número de unidades ligadas entre si que trabalham juntas para processar informação e obter resultados significativos como consequência. Estas Redes Neuronais podem ser aplicadas a modelos de classificação ou regressão contínua.

Uma Rede Neuronal pode englobar as seguintes *layers*:

- **Input Layers** - Recebe, de forma passiva, a informação inicial que entra na rede (padrões)
- **Hidden Layers** - Processamento e extração de características
- **Output Layers** - Conclui e apresenta o resultado final

Quanto maior o número de camadas, melhor a capacidade de aprendizagem da rede. Na Figura 2.7 é possível observar um diagrama simplificado de uma rede neuronal.

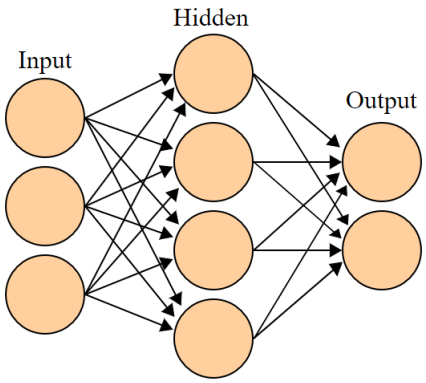

Figura 2.7: Diagrama de uma Rede Neuronal Artifical - criado por Colin M.L. Burnett

2.2.6 Avaliação dos Resultados

Um sistema de classificação deve ser capaz de conseguir prever, de forma acertada, a que classe pertence novo um objeto. Caso isto não seja possível deparamo-nos com um erro. A medida de desempenho de um classificador é obtida calculando a sua taxa de erros num conjunto de

dados que não os dados que criaram o classificador, ou seja, nos chamados dados de teste. A validação dos resultados finais irá ser feita de forma quantitativa utilizando métricas de *Data Mining* e *Machine Learning* ou de Estatística.

Matriz de Confusão

A matriz de confusão representa o número de previsões corretas e incorretas feitas pelo modelo de classificação em comparação com os resultados reais. Nesta matriz são representados quatro conceitos que posteriormente servirão para facilitar o cálculo das métricas que irão ser utilizadas na avaliação dos resultados finais: *Accuracy*, *Precision* e *Recall*.

Considerando um problema de classificação binário com classes positivo ou negativo, os quatro conceitos são:

- Verdadeiros Positivos (**VP**): número de exemplos corretamente classificados na classe dos positivos;
- Falsos Positivos (**FP**): número de exemplos incorretamente classificados como positivos;
- Verdadeiros Negativos (**VN**): número de exemplos corretamente classificados pertencendo à classe dos negativos;
- Falsos Negativos (**FN**): número de exemplos classificadas incorretamente como negativos.

Accuracy

Proporção entre os dados previstos e o seu verdadeiro valor.

$$Accuracy = \frac{VP+VN}{VP+VN+FP+FN}$$

Precision

Proporção de casos positivos que estão corretamente identificados.

$$Precision = \frac{VP}{VP+FP}$$

Recall

Proporção de casos positivos reais que estão corretamente identificados.

$$Precision = \frac{VP}{VP+FN}$$

F-Measure

A *F-Measure* (medida F) é uma medida que combina *precision* e *recall*, expressa segundo a formula:

$$F = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

AUC

A área debaixo da curva (AUC) e calculada através da área que se encontra por baixo da curva ROC. A curva ROC é uma representação gráfica que ilustra o desempenho de um sistema de classificação. Traduz um compromisso entre identificar mais exemplos positivos mas, ao mesmo tempo, aumentar o número de falsos positivos. O modelo será melhor quanto maior a sua AUC. O melhor resultado possível será 1 e no caso da AUC = 0,5 diz-se que o classificador é aleatório.

Apesar de ser possível obter estas métricas recorrendo aos cálculos demonstrados em cima, ferramentas como o *Weka*, descrito na Secção 2.2.3, facilitam o trabalho da avaliação dos resultados disponibilizando a visualização rápida dos valores de *Accuracy*, *Precision*, *F-Measure* e *AUC*, entre outros.

2.3 Trabalhos Relacionados

Existem alguns estudos na área da previsão de efeitos adversos de fármacos. No entanto, é importante referenciar o trabalho feito anteriormente sobre a *Previsão de efeitos adversos de medicamentos* por *Jéssica Namora*. Neste trabalho foram efetuadas duas abordagens ao problema.

Na primeira utilizaram-se sistemas de recomendação com recurso à base de dados ADReCS. A informação recebida pelos modelos foram os medicamentos, efeitos adversos e relação entre o par medicamento-efeito adverso. Foram utilizados os algoritmos *Slope One*, *User K-NN* e *Matrix Factorization* e foi possível obter uma *Accuracy* de 47.71% no modelo criado pelo *Matrix Factorization*. Dado à grande insatisfação dos resultados agruparam-se os dados de entrada em efeitos adversos que atuam no mesmo sistema de órgãos e foi possível então obter uma *Accuracy* de 79.52% no modelo criado pelo *Matrix Factorization*.

Na segunda abordagem os dados utilizados foram obtidos a partir de uma outra base de dados que relaciona o medicamento a um conjunto de descritores moleculares. Os algoritmos de classificação utilizados foram o *CART*, *Random Forest*, *Naive Bayes* e *SVM*. Foi obtida uma *Accuracy* de 79.02% no algoritmo *Random Forest*. Este algoritmo foi otimizado recorrendo ao método de *feature selection* a partir do cálculo da impureza de *Gini Index* e foi possível obter uma *Accuracy* de 79.49%.

Foi então possível obter uma *Accuracy* algo satisfatória em ambas as abordagens, sendo que foi necessário efetuar um pré-processamento em cada uma delas de forma a conseguir obter melhores resultados. Foi possível verificar que utilizando descritores moleculares como dados de entrada, o algoritmo de *Random Forest* foi o que conseguiu construir o modelo mais adequado.

Outros trabalhos que também mereceram atenção antes de iniciar este projeto foram ***Predicting Drugs Adverse Side-Efects using a Recommender-System*** [PCCC15] por Diogo Pinto, Pedro Costa, Rui Camacho e Vítor Santos Costa e o ***Projecto FCT "ADE - Adverse Drug Effects Detection"*** que teve como objetivo estudar a utilização de técnicas de DM para prever efeitos adversos. Este projeto teve inicio em Janeiro de 2012 e terminou em 2015 tendo como investigador principal

Vítor Santos Costa e uma equipa de investigação constituida por Ines Dutra, Rui Camacho, Nuno Fonseca, David Pages e Jesse Davis.

2.4 Resumos e Conclusões

Aliando a quimioinformática e o *data mining* possibilita a evolução da resolução do problema do estudo da interação entre fármacos e a previsão dos seus efeitos adversos.

A partir dos princípios ativos de um fármaco e dos seus *pathways* é possível melhorar a perceção de como os medicamentos atuam e interagem entre si. Estas interações podem ser benéficas ou efeitos adversos, sendo que estes efeitos adversos são muitas vezes desconhecidos inicialmente.

As propriedades de cada composto, incluindo os seus *fingerprints* e descritores moleculares podem ser obtidas a partir da sua estrutura 2D ou 3D com a ajuda de *software* como o PaDEL-Descriptor ou OpenBabel.

Para permitir que o cálculo de descritores moleculares seja possível, é necessário a recolha de informação que pode ser obtida a partir de repositórios *web*.

Existem variados repositórios, com informação muito diversificada, sendo que é necessário saber qual a informação que pretendemos obter antes de escolher que repositório devemos utilizar. Com o objetivo de obter informação relativa aos medicamentos e aos seus efeitos adversos conhecidos podemos então aceder a vários repositórios, tais como: *openFDA, PubChem, RxNav, ADReCS, ICD, THIN e MedRA*. Para obter informação de biologia e interações moleculares podem-se utilizar os seguintes repositórios: *ChEBI, DrugBank, SIDER, STITCH* e *KEGG*.

Após obter a informação pretendida através dos repositórios e descritores moleculares podemos utilizar o *data mining* para chegar às conclusões necessárias. Esta informação tem que ser organizada, integrada, selecionada e transformada de forma a que seja possível aplicar algoritmos de *data mining* obtendo resultados que depois vão ser avaliados e, caso sejam satisfatórios, o conhecimento obtido através destes poderá ser utilizado para ser aplicado no seu uso pretendido, a previsão de efeitos adversos na interação entre fármacos.

Existem diversas ferramentas de *data mining* que possibilitam a aplicação de tarefas de previsão, tais como o *Weka*, o *R*, o *RapidMiner* e o *KNIME*. Dentro destas tarefas de previsão, temos então os algoritmos de classificação. A função destes algoritmos é de encontrar um grupo de modelos ou funções que irão descrever ou distinguir um conjunto de classes ou conceitos num *dataset* para permitir o uso destes modelos na previsão das classes ou conceitos em falta num determinado objeto. Alguns destes algoritmos são o *SVM* que é um classificador linear binário não probabilístico, o *Random Forest* que é um tipo de *ensemble learning*, o *k-NN* que é um tipo de *lazy learning*, o *J48*, o *AdaBoost* que combina vários algoritmos de aprendizagem simples para criar uma previsão de alta precisão e a Rede Neuronal Artificial (*Multilayer Perceptron*) que é uma técnica de processamento de informação que se assemelha ao processamento de informação feito pelo cérebro.

Após a aplicação destes algoritmos é necessário avaliar os resultados obtidos. Ferramentas com o *Weka* permitem a visualização de várias métricas de avaliação importantes, tais como a

Accuracy, a *Precision*, a *F-Measure* e a *AUC*. A partir destas métricas podemos concluir se o modelo criado é bom o suficiente ou se os resultados não são satisfatórios.

Capítulo 3

Solução Proposta

Neste capítulo é detalhado o tratamento inicial da informação e metodologia a aplicar bem como a escolha dos algoritmos de pré-processamento e de *data mining* a utilizar no decorrer deste projeto.

Como mencionado no Capítulo 1 o objetivo principal é tentar prever os efeitos adversos que advém da interação entre medicamentos para criar uma forma de previsão que seja mais precisa e menos dispendiosa que as técnicas mais utilizadas hoje em dia, que se baseiam em testes utilizando uma pequena amostra da população que muitas vezes acaba por não ser representativa do consumidor final, tornando mais difícil de obter resultados rigorosos.

Os repositório selecionados para a obtenção da informação inicial foram o *openFDA* e o *PubChem* descritos na Secção 2.1.2.1, tendo sido também utilizado o *ChEBI* (descrito na Secção 2.1.2.2) para obtenção de informação sobre as interações moleculares conhecidas e utilização de informação das ontologias Químicas/Farmacológicas.

As ferramentas escolhidas para a elaboração deste projeto foram o *PaDEL* (descrito na Secção 2.1.3) para obtenção dos descritores moleculares e o Weka (descrito na Secção 2.2.3 para as tarefas de data mining).

3.1 Criação dos *datasets*

3.1.1 Organização da base de dados

Primeiramente foram obtidos, a partir do openFDA, ficheiros JSON que continham episódios de efeitos adversos associados a vários fármacos descritos por qualquer individuo em qualquer país do mundo, através de um portal que o openFDA tem para o efeito. Os episódios estão datados, pertencendo a vários anos. No nosso estudo utilizamos os episódios entre 2004 e 2015. Em cada ano os episódios estão organizados por quartis.

Dentro de cada episódio são fornecidas várias informações, sendo as principais a idade e género do indivíduo, o nome dos fármacos associados a cada episódio e os seus RxCUIs (*RxNorm Concept Unique Identifier*), que são os números de identificação únicos que descrevem o conceito

semântico sobre cada produto farmacêutico, incluindo os seus ingredientes, pontos fortes e formas de dosagem.

Para poder utilizar toda esta informação neste projeto foi necessário recolher os dados mais relevantes e colocá-los numa base de dados de forma a posteriormente poderem ser mais facilmente acedidos, a partir de APIs e outras ferramentas e conseguir construir ficheiros ARFF para assim aplicar os algoritmos do Weka.

Para a recolha inicial de informação foi criado um *parser* em Python para os ficheiros JSON originais, como é possível ver na Secção A.1 do Apêndice. As informações recolhidas pelo *parser* foram guardadas nas tabelas de uma Base de Dados representadas na Figura 3.1, os nomes de cada fármaco foram guardados na tabela *substances* e os RxCUIs foram colocados na tabela *subrxcui* na mesma base de dados.

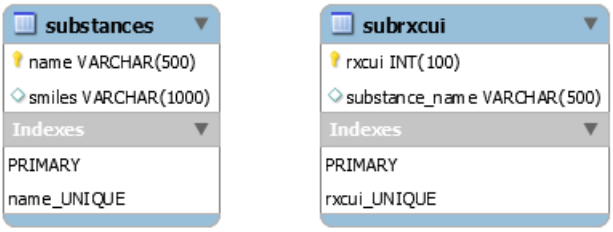

Figura 3.1: Esquema das tabelas de 1 Base de Dados *substances* e *subrxcui*

A partir daí foi possível utilizar estas informações retiradas dos ficheiros JSON do OpenFDA e utilizar a API do PubChem [1] para obter mais informações acerca de cada fármaco. As informações adicionais sobre fármacos incluem as seguintes:

- **CID** - *Compound IDentifier*: Obtido a partir do nome de cada fármaco com o propósito de ser utilizado para obter as restantes informações que apenas se consegue adquirir utilizando este identificador;
- Canonical **SMILES**[2] - *Simplified Molecular Input Line Entry System*: É uma *string* única que codifica a estrutura do composto, gerado por um algoritmo de padronização. Esta informação será utilizada pela ferramenta PaDEL para obter os descritores moleculares de cada composto.

Foi ainda obtida informação acerca das **interações de alta prioridade entre fármacos** que já sejam conhecidas a partir da API do RxNav [3], para conseguir esta informação foram utilizados os identificadores RxCUI e todas as interações foram colocadas na base de dados na tabela *interactions*.

Para aceder às propriedades da **ontologia** de cada compostos foi utilizada uma API do ChEBI, a libChEBI [4]. Primeiramente foi obtido o ChEBi ID a partir da API do PubChem, sendo que estes

[1] http://pubchemdocs.ncbi.nlm.nih.gov/pug-rest
[2] Esquema de representação da estrutura de uma molécula
[3] https://rxnav.nlm.nih.gov/InteractionAPIREST.html
[4] https://github.com/libChEBI/libChEBIpy

foram utilizados para obter, na Ontologia, informação adicional para cada composto. Toda esta informação foi também colocada na base de dados.

Na Figura 3.2 podemos ver a estrutura de todas as tabelas da Base de Dados criada para guardar toda a informação obtida nesta fase.

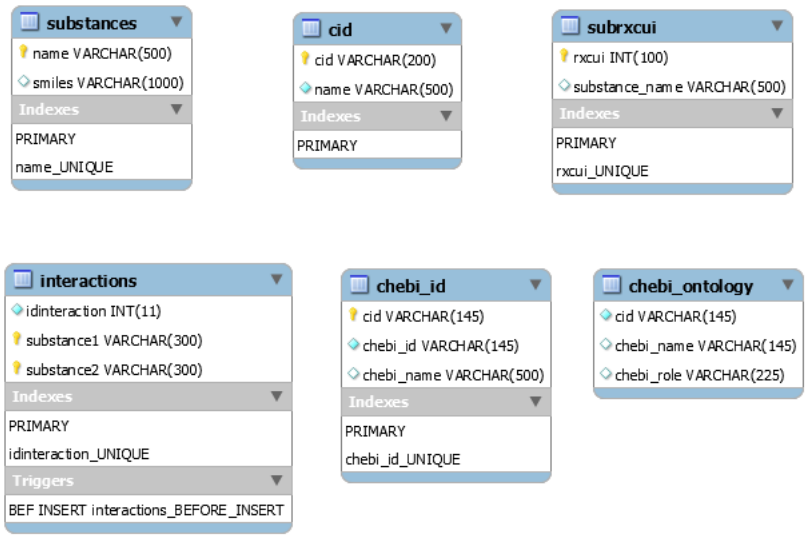

Figura 3.2: Estrutura de todas as tabelas contidas na Base de Dados

3.1.2 Descritores Moleculares

Os descritores moleculares foram obtidos a partir da ferramenta PaDEL. Para utilizar esta ferramenta foi necessário ir à base de dados obter os SMILES de todos os compostos e colocá-los em ficheiros com o nome correspondente ao composto em causa. A partir daí foram criados os descritores 1D e 2D a partir desses ficheiros. Na Figura 3.3 é possível ver esta ferramenta a calcular os descritores moleculares.

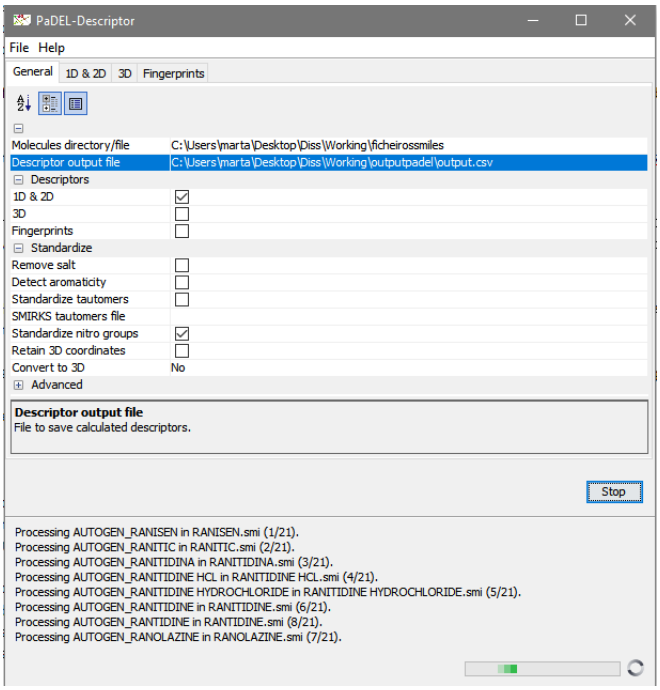

Figura 3.3: PaDEL

Após o cálculo dos descritores moleculares, o PaDEL disponibilizou esta informação num ficheiro *csv*. Foi decidido não se calcular os descritores 3D com o PaDEL, pois o elevado número de compostos e a complexidade de alguns SMILES faziam com que a ferramenta demorasse um tempo inaceitável.

3.1.3 *Datasets*

A partir do ficheiro criado pelo PaDEL foi possível obter a base do *dataset* inicial. Neste ficheiro *csv* inicial em cada linha é representada uma molécula com todos os atributos que o PaDEL calculou mas para ser possível estudar as interações entre as moléculas teve que se alterar cada uma destas linhas para apresentar um par de moléculas com a classe correspondente que especifica se há ou não uma interação conhecida entre essas moléculas. Isto foi possível criando uma lista de pares de interações conhecidas entre moléculas e depois uma lista de pares de moléculas que não tinham interação conhecida e após isso substituir os nomes das moléculas em cada par pelos seus descritores, obtendo assim o *dataset*. A informação sobre as interações conhecidas foi retirada da tabela *interactions* descrita na Figura 3.4.

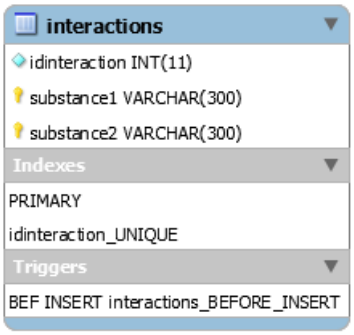

Figura 3.4: Esquema da Tabela *interactions*

Como se pode ver na Figura 3.5, a partir deste *dataset* foram obtidos 5 *datasets* que foram separados entre *training* e *test* set sendo que 80% foi para o *training set* e os restantes 20% foram para o *test set*. O propósito é ser possível calcular uma média e desvio padrão do desempenho de cada algoritmo e poder depois realizar testes de significância estatística sobre o desempenho de diferentes algoritmos.

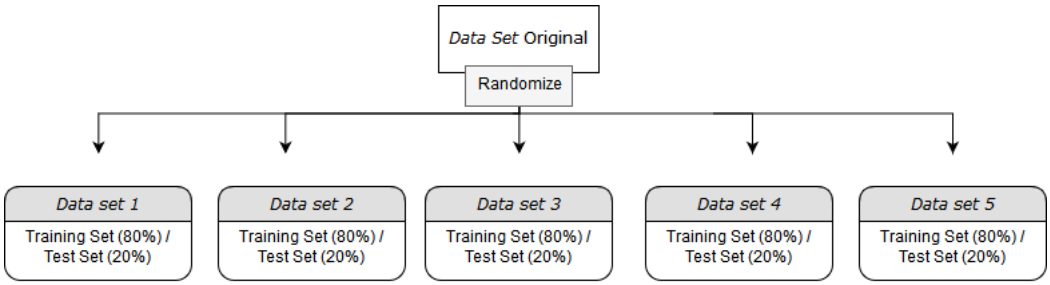

Figura 3.5: Criação de 5 *datasets* com os mesmos exemplos do *dataset* original mas com divisão entre treino e teste diferentes (gerados aleatoriamente).

3.2 Pré-processamento e Processamento dos *datasets*

Para ser possível criar um modelo que preveja as interações entre moléculas, vários algoritmos de *data mining* foram utilizados bem como métodos de filtragem dos 5 *datasets* para tentar aumentar a *performance* dos algoritmos.

Foram escolhidos vários algoritmos a utilizar sendo que cada algoritmo foi analisado para perceber que parâmetros poderiam ser variados para tentar melhorar o desempenho de cada um deles.

Primeiramente foram aplicados os algoritmos nos *datasets* sem que estes tivessem qualquer pré-processamento. De seguida, foram utilizados os métodos de Normalização, *Feature Selection* e *Attribute Enrichment*, divididos por várias fases. O método de *Feature Selection* foi aplicado de 3 maneiras diferentes.

3.2.1 Sem pré-processamento

Primeiramente estes algoritmos foram aplicados nos *datasets* sem qualquer filtragem, apenas com a separação inicial entre *training* e *test set*. Como é possível ver no esquema da Figura 3.6, para analisar os resultados desta primeira experiência foi calculada a média e o desvio padrão da melhor *run* de cada algoritmo.

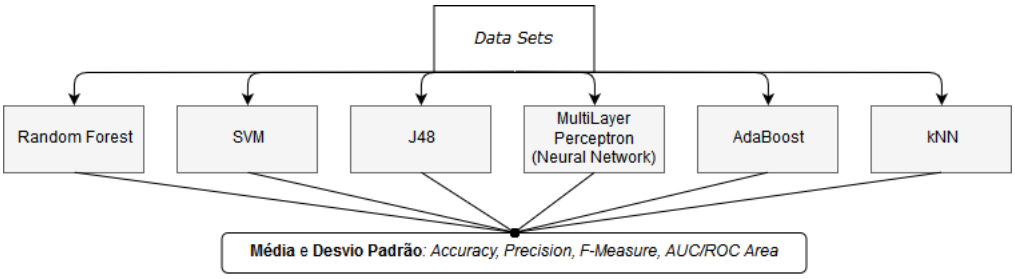

Figura 3.6: Aplicação dos algoritmos escolhidos nos 5 *datasets* base

3.2.2 Normalização

Para tentar obter resultados mais favoráveis os *datasets* foram normalizados a partir do filtro *Standardize* que padroniza todos os atributos numéricos num determinado *dataset* para terem uma distribuição com média zero e de unidade (expecto o atributo de classe) como demonstrado na Figura 3.7.

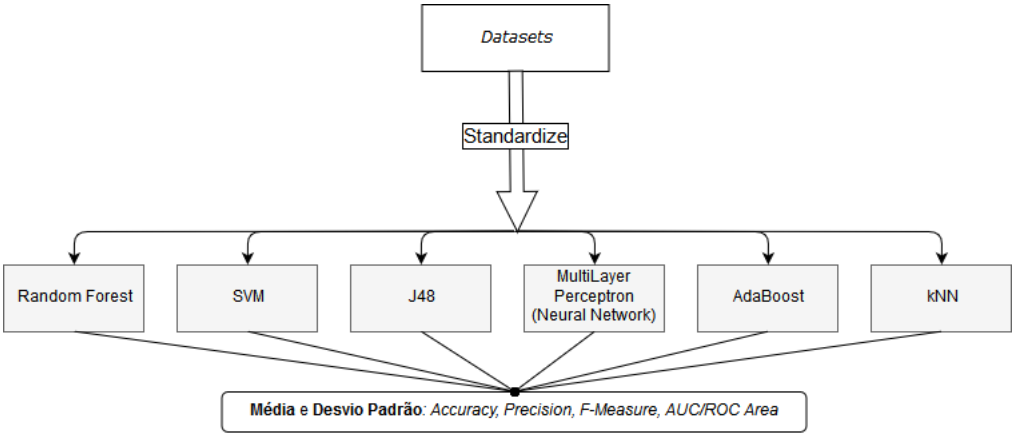

Figura 3.7: Normalização e aplicação dos algoritmos escolhidos nos 5 *datasets*

3.2.3 *Feature Selection*

Posteriormente à normalização dos *datasets*, e para obter um desempenho superior, foi realizada uma operação de *feature selection* (selecção de atributos). É possível ver uma representação deste método na Figura 3.8. O *Feature Selection* foi realizado num dos 5 *datasets* obtidos inicialmente, após a sua normalização, e a partir do seu *training set* fez-se novamente uma separação de

80%/20% para obter novos *sets* de *train* e *test*. Após obter estes novos *sets*, foi utilizado o *training set* nos métodos de *feature selection*.

O primeiro processo foi realizado usando um *Attribute Evaluator* (avaliador de atributos) e com um *Search Method* (método de procura) associado. Foi escolhido o *attribute evaluator* ***CorrelationAttributeEval*** com o *search method* ***Ranker***. Com este método foi obtida uma lista de atributos, ordenados pelo *ranker*, cada um com uma pontuação sendo que os que têm maior pontuação são aqueles que têm maior importância. O Weka disponibilizou também uma lista apenas com os índices de cada atributo ordenados por valor de correlação. Selecionaram-se os índices dos atributos que tinham o valor de correlação 0 (zero) e utilizando o filtro *Remove* estes atributos foram removidos dos 5 *datasets*.

Seguidamente foram também usados mais dois métodos de *feature selection*. No primeiro foi feito um corte drástico, onde foi cortado um grande número de atributos, restando apenas o número de atributos equivalente ao número de instâncias desse *dataset*.

O procedimento seguinte fez-se utilizando o algoritmo J48. Correu-se este algoritmo, com o número mínimo de instâncias (m) igual a 25, nos 5 *datasets* normalizados e foi feita uma lista de todos os atributos que apareceram no conjunto de árvores que foi possível obter. Esta lista de atributos foi então utilizada no *feature selection* onde apenas foram mantidos estes atributos, eliminando todos os outros. Esta abordagem tem ainda como objetivo conseguir obter informação sobre quais os melhores atributos (para um algoritmo como o J48).

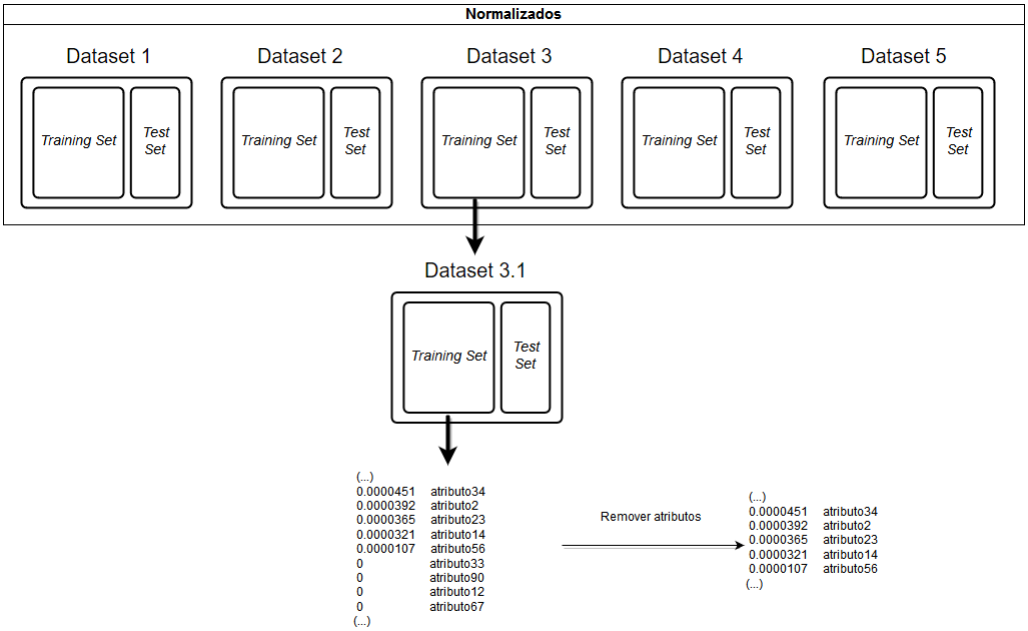

Figura 3.8: Feature Selection

3.2.4 *Attribute Enrichment*

Após a remoção dos atributos classificados com menor importância sucedeu-se o passo de *attribute enrichment*. A partir da base de dados foi possível obter algumas propriedades extra

relativas a cada molécula, com esta informação adicionaram-se novos atributos a cada molécula. Estas propriedades foram obtidas a partir do ChEBI, onde cada molécula pode ter associada 3 propriedades:

- Função Química: classifica as entidades com base nas suas funções dentro de um contexto químico (exemplo: inibidor);
- Função Biológica : classifica as entidades com base no seu papel num contexto biológico (exemplo: antibiótico, agente antiviral, hormonal)
- Aplicação: classifica com base no uso dado pelos seres humanos (exemplo: medicamentos anti-reumáticos).

Ao obter os valores de todas as propriedades da ontologia de cada composto foram então adicionados novos atributos em que cada atributo era uma propriedade da ontologia. Caso essa propriedade fosse pertencente à ontologia desse composto o valor desse atributo será "1", caso contrário o valor será "0". Foram então adicionados 498 novos atributos dado que cada instância tinha um par de compostos.

Capítulo 4

Caso de Estudo

Neste capítulo são descritas todas as experiências feitas no Weka e também se apresenta e discute os seus respetivos resultados.

4.1 Dados e Algoritmos

4.1.1 Composição dos *datasets*

O *dataset* original contêm 518 instâncias e 2889 atributos no total. Os 5 *datasets* criados a partir do *dataset* inicial tinha um *training set* com 414 instâncias e um *test set* com 104 instâncias. A classe que classifica se há uma interação conhecida entre as moléculas, de forma positiva ou negativa foi dividida de forma balanceada em todos os *datasets*, existindo no *training set* 207 positivos e 207 negativos e no *test set*, 52 positivos e 52 negativos em cada grupo de *datasets*.

4.1.2 Variação de parâmetros

Foram utilizados os algoritmos listados na Tabela 4.1. A Tabela mostra também os valores de cada parâmetro que foram experimentados.

Tabela 4.1: Algoritmos utilizados e respetivos parâmetros que foram sujeitos a sintonização

Algoritmo	***Atributos***	***Valores***
Random Forest	numIterations (Number of Trees)	*100, 500, 1000*
SVM	kernelType	*Radial Basis, Polynomial, Sigmoid*
J48	M (Minimum Number of Objects)	*2, 10, 20*
Multilayer Perceptron[1]	hiddenLayers	*1*
AdaBoost	classifier	*Decision Stump, J48*
	numIterations	*10, 100*
k-NN	k	*1, 7*
	distanceFunction	*Euclidean, Manhattan*

4.2 Experiências e Resultados

Como explicado no Capítulo 3 primeiramente foram utilizados 5 *datasets* provenientes do *dataset* obtido a partir do PaDEL e das interações conhecidas. Cada *dataset* foi depois dividido entre *training* e *test set* para poderem ser aplicados os algoritmos escolhidos previamente. É sabido que um classificador que não tenha qualquer trabalho e apenas diga que todos os resultados são positivos ou todos os resultados são negativos irá ter sempre aproximadamente 50% de probabilidade de acertar. Dito isto, o resultado de referência para a *Accuracy* de todas as experiências será a percentagem da classe maioritária, ou seja, aproximadamente 50%. Se não for possível obter uma *Accuracy* maior que 50% então o esforço feito pelos algoritmos será desnecessário.

Foram utilizados os algoritmos de *Random Forest*, *SVM*, *J48*, *AdaBoost*, *k-NN* e *Multilayer Perceptron* (Rede Neuronal). Este último apenas foi utilizado nas experiências 4 e 5 pois nas restantes não foi possível obter resultados palpáveis devido ao elevado número de atributos.

Na Tabela 4.2 podem ser observados os resultados da experiência 1, esta experiência foi feita com os 5 *datasets* obtidos a partir do *dataset* inicial, não sendo efetuado nenhum pré-processamento. Os algoritmos com melhor desempenho foram o *Random Forest* e o *AdaBoost* e o algoritmo com pior desempenho foi o *SVM*.

Para a Experiência 2 foi feita a normalização dos valores de cada atributo, utilizando o filtro *Standardize* do Weka. Como se pode ver na Tabela 4.3 o *Random Forest* foi novamente o algoritmo com melhor desempenho e o *J48* foi o algoritmo com os piores resultados.

Na Experiência 3 foi efetuado um pré-processamento de *Feature Selection* onde a partir do seletor *CorrelationAttributeEval* foi obtida a lista completa de atributos organizada por *rank* onde os atributos com rank = 0 foram eliminados, restando 2269 atributos. Os resultados, descritos na Tabela 4.4 mostram que o melhor algoritmo foi o *Random Forest*.

Na Experiência 4 foi efetuado um corte drástico nos atributos mantendo apenas 332 atributos. Os resultados, representados na Tabela 4.5, mostram que o algoritmo com maior *Accuracy* foi o *Random Forest*.

Na Experiência 5 efetuou-se um corte também bastante drástico, onde se mantiveram apenas os atributos utilizados pelo algoritmo J48 nos 5 *datasets* iniciais. Na Figura 4.1 podemos ver as árvores geradas pelo algoritmo J48.

Dataset	Árvore gerada pelo algoritmo J48
Dataset 1	GATS2p_1 <= 1.00773 | GATS6s_1 <= 0.550358: pos (28.0/13.0) | GATS6s_1 > 0.550358: negs (132.0/11.0) GATS2p_1 > 1.00773 | VE1_Dt_1 <= 0.231679 | | ATSC8e_1 <= 0.017924: pos (192.0/21.0) | | ATSC8e_1 > 0.017924: negs (25.0/6.0) | VE1_Dt_1 > 0.231679: negs (37.0/4.0)
Dataset 2	GATS2p_1 <= 1.00773: negs (159.0/24.0) GATS2p_1 > 1.00773 | BIC3_1 <= 0.812898: negs (26.0) | BIC3_1 > 0.812898 | | MATS4c_1 <= -0.036985: negs (31.0/8.0) | | MATS4c_1 > -0.036985: pos (198.0/23.0)
Dataset 3	GATS2p_1 <= 1.005647 | AATSC6m_1 <= 2.397802: negs (110.0/5.0) | AATSC6m_1 > 2.397802 | | ATSC5m_2 <= -110.905736: pos (25.0/9.0) | | ATSC5m_2 > -110.905736: negs (26.0/4.0) GATS2p_1 > 1.005647 | ATSC4i_1 <= -14.056037: negs (29.0/1.0) | ATSC4i_1 > -14.056037 | | ATSC8s_1 <= 0.532544 | | | AATS5i_1 <= 160.749574: negs (26.0/11.0) | | | AATS5i_1 > 160.749574: pos (168.0/7.0) | | ATSC8s_1 > 0.532544: negs (30.0/9.0)
Dataset 4	GATS2p_1 <= 0.995796: negs (137.0/18.0) GATS2p_1 > 0.995796 | VE1_Dt_1 <= 0.231679 | | SIC3_1 <= 0.848435: negs (42.0/13.0) | | SIC3_1 > 0.848435: pos (193.0/22.0) | VE1_Dt_1 > 0.231679: negs (42.0/5.0)
Dataset 5	AATSC7i_1 <= -0.178185: negs (50.0) AATSC7i_1 > -0.178185 | BIC2_1 <= 0.782723 | | MATS3v_1 <= -0.111651: negs (45.0/6.0) | | MATS3v_1 > -0.111651 | | | GATS5s_1 <= 1.228661 | | | | AATSC1m_1 <= 1.796633 | | | | | MATS4s_1 <= -0.063043: negs (28.0/13.0) | | | | | MATS4s_1 > -0.063043: pos (182.0/11.0) | | | | AATSC1m_1 > 1.796633: negs (26.0/7.0) | | | GATS5s_1 > 1.228661: negs (28.0/5.0) | BIC2_1 > 0.782723: negs (55.0/5.0)

Figura 4.1: Árvores geradas pelo algoritmo J48

A partir destas árvores foi possível selecionar 19 atributos que podemos ver detalhados em baixo (*_1* significa que o atributo pertence ao primeiro elemento do par de moléculas e *_2* significa que pertence ao segundo elemento do par):

- GATS2p_1 (*Geary autocorrelation of lag 2 weighted by polarizability*)
- GATS6s_1 (*Geary autocorrelation of lag 6 weighted by I-state*)
- VE1_Dt_1 (*Coefficient sum of the last eigenvector from detour matrix*)
- ATSC8e_1 (*Centred Broto-Moreau autocorrelation of lag 8 weighted by Sanderson electronegativity*)
- BIC3_1 (*Bond Information Content index (neighborhood symmetry of 3-order)*)
- MATS4c_1 (*Moran autocorrelation - lag 4 / weighted by charges*)
- AATSC6m_1 (*Average Centred Broto-Moreau autocorrelation of lag 6 weighted by mass*)
- ATSC5m_2 (*Centred Broto-Moreau autocorrelation of lag 5 weighted by mass*)
- ATSC4i_1 (*Centred Broto-Moreau autocorrelation of lag 4 weighted by ionization potential*)
- ATSC8s_1 (*Centred Broto-Moreau autocorrelation of lag 8 weighted by I-state*)
- AATS5i_1 (*Average Broto-Moreau autocorrelation - lag 5 / weighted by first ionization potential*)
- SIC3_1 (*Structural Information Content index (neighborhood symmetry of 3-order)*)
- AATSC7i_1 (*Average centered Broto-Moreau autocorrelation - lag 7 / weighted by first ionization potential*)
- BIC2_1 (*Bond Information Content index (neighborhood symmetry of 2-order)*)
- MATS3v_1 (*Moran autocorrelation of lag 3 weighted by van der Waals volume*)
- GATS5s_1 (*Geary autocorrelation of lag 5 weighted by I-state*)
- AATSC1m_1 (*Average centered Broto-Moreau autocorrelation - lag 1 / weighted by mass*)
- MATS4s_1 (*Moran autocorrelation of lag 4 weighted by I-state*)

Após a remoção dos atributos e o processamento dos algoritmos foi possível obter os resultados presentes na Tabela 4.6 onde o algoritmo com melhor desempenho foi o *AdaBoost*.

Na Experiência 6, onde se efetuou um *Attribute Enrichment*, teve um grande aumento do número do atributos passando para 3387. Foram obtidos os resultados descritos na Tabela 4.7 onde se pode observar que os melhores algoritmos foram o *Random Forest* e o *AdaBoost*. Apesar do *Random Forest* ter obtido uma *Accuracy* média maior, o grande desvio-padrão da *Accuracy* do *AdaBoost* (3.999%) leva a querer que estes algoritmos poderão ter tido um desempenho semelhante.

Tabela 4.2: Resultados Experiência 1 - Sem pré-processamento

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	0.90 ± 0.04	0.90 ± 0.04	0.95 ± 0.02	90.20 ± 3.62
SVM	0.76 ± 0.04	0.76 ± 0.04	0.76 ± 0.04	75.80 ± 3.56
J48	0.86 ± 0.03	0.86 ± 0.03	0.87 ± 0.02	85.80 ± 2.67
AdaBoost	0.90 ± 0.03	0.90 ± 0.03	0.93 ± 0.03	90.20 ± 2.99
k-NN	0.82 ± 0.04	0.81 ± 0.04	0.84 ± 0.07	81.40 ± 3.75

Tabela 4.3: Resultados Experiência 2 - Normalização

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	0.88 ± 0.05	0.87 ± 0.05	0.93 ± 0.03	87.10 ± 5.34
SVM	0.78 ± 0.01	0.77 ± 0.03	0.78 ± 0.02	77.50 ± 2.21
J48	0.71 ± 0.11	0.68 ± 0.10	0.71 ± 0.08	68.80 ± 9.58
AdaBoost	0.86 ± 0.04	0.85 ± 0.04	0.90 ± 0.03	85.00 ± 4.11
k-NN	0.82 ± 0.04	0.81 ± 0.04	0.84 ± 0.05	80.80 ± 3.91

Tabela 4.4: Resultados Experiência 3 - Feature Selection: Corte dos atributos com rank=0

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	0.89 ± 0.06	0.87 ± 0.08	0.94 ± 0.03	87.50 ± 7.26
SVM	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	82.70 ± 1.36
J48	0.78 ± 0.07	0.75 ± 0.07	0.77 ± 0.08	75.80 ± 7.18
AdaBoost	0.85 ± 0.05	0.83 ± 0.06	0.89 ± 0.04	83.30 ± 5.42
k-NN	0.82 ± 0.04	0.81 ± 0.04	0.84 ± 0.05	80.80 ± 3.91

Tabela 4.5: Resultados Experiência 4 - Feature Selection: Corte Drástico (número de atributos = número de instâncias do *dataset*).

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	0.89 ± 0.02	0.88 ± 0.02	0.93 ± 0.02	88.50 ± 2.26
SVM	0.86 ± 0.03	0.86 ± 0.03	0.86 ± 0.03	86.00 ± 2.93
J48	0.81 ± 0.05	0.79 ± 0.07	0.80 ± 0.08	79.00 ± 6.81
AdaBoost	0.85 ± 0.04	0.84 ± 0.03	0.89 ± 0.02	84.20 ± 3.09
k-NN	0.87 ± 0.03	0.86 ± 0.03	0.87 ± 0.03	86.20 ± 3.30
Multilayer Perceptron	0.87 ± 0.02	0.86 ± 0.02	0.87 ± 0.03	86.00 ± 2.41

Tabela 4.6: Resultados Experiência 5 - Feature Selection: Corte Nós J48.

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	0.87 ± 0.01	0.87 ± 0.01	0.91 ± 0.02	86.90 ± 1.10
SVM	0.77 ± 0.04	0.58 ± 0.01	0.64 ± 0.01	63.50 ± 1.36
J48	0.88 ± 0.03	0.87 ± 0.02	0.87 ± 0.03	87.30 ± 2.39
AdaBoost	0.89 ± 0.01	0.87 ± 0.01	0.91 ± 0.03	88.70 ± 1.05
k-NN	0.86 ± 0.01	0.86 ± 0.01	0.87 ± 0.02	86.20 ± 0.09
Multilayer Perceptron	0.87 ± 0.02	0.86 ± 0.02	0.87 ± 0.03	86.00 ± 2.41

Tabela 4.7: Resultados Experiência 6 - Attribute Enrichment

Algoritmo	*Precision*	*F-Measure*	AUC / ROC Area	Accuracy %
Random Forest	**0.93 ± 0.02**	**0.93 ± 0.02**	**0.97 ± 0.01**	**93.10 ± 1.23**
SVM	0.77 ± 0.03	0.77 ± 0.03	0.77 ± 0.03	76.70 ± 2.67
J48	0.91 ± 0.03	0.90 ± 0.03	0.93 ± 0.02	90.40 ± 2.80
AdaBoost	**0.91 ± 0.03**	**0.91 ± 0.04**	**0.96 ± 0.02**	**90.80 ± 3.99**
k-NN	0.84 ± 0.03	0.83 ± 0.04	0.87 ± 0.03	82.90 ± 3.62

4.2.1 Discussão dos resultados

Na Experiência 1 foram obtidos resultados bastante satisfatórios. Nos algoritmos de *Random Forest* e *AdaBoost* foi obtida uma *Accuracy* de 90.2% em ambos. Apesar do *AdaBoost* ter um desvio padrão menor na *Accuracy*, o *Random Forest* atingiu melhores resultados ao nível da *Precision* e *AUC*. O algoritmo com pior resultado foi o SVM com apenas 75.8% de *Accuracy*.

Na Experiência 2 onde foi feito um pré-processamento de normalização, houve uma melhoria no algoritmo *SVM* com um aumento na *Accuracy* de 1.7%, mas nos restantes os resultados foram menos satisfatórios. O algoritmo com melhores resultados foi o de *Random Forest* com uma *Accuracy* de 87.1% e o pior algoritmo foi o *J48* com uma *Accuracy* de 68.8%.

Nas experiências onde foi efetuado um pré-processamento de *feature selection* os resultados não foram muito satisfatórios em comparação com a Experiência 1. Na Experiência 3 os resultados melhoraram em relação à Experiência 2, excetuando no algoritmo *AdaBoost* que obteve piores resultados e o algoritmo *k-NN* que se manteve igual. Nesta experiência, o algoritmo com melhores resultados foi o *Random Forest* com uma *Accuracy* de 87.5% e o pior foi o *J48* com 75.8%.

Na Experiência 4 o melhor algoritmo foi também o *Random Forest* com uma *Accuracy* de 88.5%. Houve também um aumento significativo em comparação com as experiências 2 e 3 no algoritmo *SVM*, mas não foi suficiente.

Na Experiência 5 o algoritmo com melhor desempenho AdaBoost com uma Accuracy de 88.7%.

Na Experiência 6 foi efetuado um *Attribute Enrichemnt* onde foram adicionadas as propriedades químicas, biológicas e aplicações provenientes da ontologia ChEBI. Foi possível verificar um aumento no desempenho da maior parte dos algoritmos, tendo o algoritmo *Random Forest* obtido um novo máximo com 93.1% de *Accuracy*. Os algoritmos *J48* e *AdaBoost* conseguiram obter também resultados satisfatórios com uma *Accuracy* de 90.4% e 90.8% respetivamente. Apesar do *Random Forest* ter obtido a *Accuracy* média mais elevada, o algoritmo *AdaBoost* obteve um grande desvio-padrão (3.999%) e dado a sua *Accuracy* estar próxima de 93.1% é possível que estes algoritmos sejam semelhantes.

Pode-se então dizer que, para este *dataset*, os pré-processamentos Normalização e *Feature Selection* não trouxeram melhorias no desempenho dos algoritmos e também que os cortes mais drásticos não provocavam grandes diferenças, sendo até melhores para o desempenho dos algoritmos do que o corte por *rank*. Por outro lado, conclui-se que o *Attribute Enrichment* foi um passo importante para a obtenção de resultados mais satisfatórios pois os melhores resultados foram os da Experiência 6 com os algoritmos de *Random Forest* onde se obteve uma Precision de 0.928, um F-Measure de 0.927, uma AUC de 0.973 e uma Accuracy de 93.1% $\pm$ 1.227% e o algoritmo *AdaBoost* com uma *Precision* de 0.914, um *F-Measure* de 0.907, um *AUC* de 0.955 e uma *Accuracy* de 90.8% $\pm$ 3.999 %

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Satisfação dos Objetivos

Hoje em dia, a toma de vários medicamentos em simultâneo é uma prática cada vez mais comum, e apesar de se saber que esta ação pode levar a que os medicamentos interajam entre eles de forma perigosa, resultando em efeitos adversos, muitos destes efeitos não são conhecidos.

Este trabalho focou-se no estudo da interação entre fármacos e previsão dos seus efeitos adversos através do *Data Mining*. Este estudo é importante pois, atualmente, muita da informação obtida sobre os efeitos adversos não é suficiente dado que a forma de obter esta informação foca-se principalmente em utilizar um grupo de teste constituído por pacientes que formam apenas uma pequena amostra da população e que acaba por não demonstrar uma verdadeira representação do consumidor final.

Foi necessário iniciar este trabalho com a criação de *datasets*. Estes *datasets* foram criados a partir de episódios de efeitos adversos na toma dos medicamentos que foram obtidos a partir do OpenFDA. Estes episódios relatavam que medicamentos foram tomados, sendo que toda a informação relevante foi retirada a partir de um *parser* e foi colocada numa base de dados. Posteriormente, a partir do conhecimento já disponível, adquiriram-se mais informações acerca de cada um dos compostos relatados nos ficheiros OpenFDA, como o SMILES e as interações conhecidas de cada um desses compostos e disponíveis na API do RxNav. Foi também criada uma lista de interações negativas, ou seja, compostos que não teriam interações conhecidas no RxNav.

A partir do SMILES foi possível calcular os descritores moleculares no PaDEL e com esses descritores, alterar a lista de interações (positivas ou negativas), onde em vez de apresentar o nome do composto apresentaria agora o seu descritor molecular. Com isto foi possível criar o *dataset* inicial.

A partir do *dataset* foram criados 5 *datasets* com um *training* e um *test set* que, apesar de conterem todos a mesma informação, cada um deles apresentava uma ordem diferente.

Foram então realizadas seis experiências: primeiramente aplicaram-se os algoritmos aos 5 *datasets* iniciais seguindo-se de uma normalização destes *datasets* e aplicaram-se os algoritmos de novo. Após a normalização os *datasets* sofreram três métodos de *feature selection* diferentes (Corte por Rank, Corte Drástico com número de atributos igual ao número de instâncias e Corte Drástico mantendo os atributos utilizados na criação das árvores no algoritmo J48) e em cada um destes métodos foram aplicados novamente os algoritmos. Finalmente foi aplicado o método de *attribute enrichment* onde se acrescentaram as ontologias de cada molécula e aplicaram-se os algoritmos novamente. Os algoritmos utilizados foram *Random Forest*, *SVM*, *J48*, *Multilayer Perceptron*, *AdaBoost* e *k-NN*.

Após feitas as experiências foram analisados os resultados, sendo que as métricas utilizadas para medir a satisfação dos objetivos foram a *Precision*, a *F-Measure*, a *AUC* ou *ROC Area* e a *Accuracy* tendo sido feita a média e o desvio padrão de cada um deles.

Pode-se então concluir que os resultados obtidos no decorrer deste trabalho foram bastante convincentes, ainda assim fica a ideia que é necessário melhorar o pré-processamento dos *datasets* para tentar obter melhores conclusões dado que os pré-processamentos de *Feature Selection* e de Normalização neste trabalho não melhoraram o desempenho dos algoritmos como esperado. Em contrapartida, o *Attribute Enrichment* resultou numa melhoria algo significativa, tendo sido uma etapa decisiva na construção de um modelo para prever os efeitos adversos provenientes da interação entre medicamentos.

Com os resultados obtidos a partir das experiências realizadas foi possível perceber que os modelos criados pelos algoritmos *Random Forest* e *AdaBoost* seriam os melhores a aplicar a estes *datasets* e que as ontologias são uma informação importante no estudo das interações entre os medicamentos.

5.2 Trabalho Futuro

O trabalho necessário a realizar no futuro passará então por testar mais métodos para melhorar o desempenho dos algoritmos variando os tipos de pré-processamento, os algoritmos utilizados e a variação respetiva dos parâmetros. Para além disso, também se poderá criar *datasets* com mais entradas dado que este *dataset* não tinha uma grande dimensão.

Como trabalho futuro seria também interessante averiguar se os modelos construídos conseguem prever (e contribuir para uma explicação, no caso de modelos simbólicos) para possíveis interações entre medicamentos tomados pelas pessoas que reportaram episódios adversos no repositório OpenFDA e cuja interação não é ainda conhecida na literatura.

Referências

[ABMA17] Vinicius Alves, Rodolpho Braga, Eugene Muratov e Carolina Andrade. Quimioinformática: Uma introdução. *Química Nova*, 2017. URL: https://doi.org/10.21577/0100-4042.20170145, doi:10.21577/0100-4042.20170145.

[BCD+07] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel e Bernd Wiswedel. KNIME: The Konstanz Information Miner. In *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer, 2007.

[BPZM13] Olivier Bodenreider, Lee Peters, Kelly Zeng e Jonathan Mortensen. RxNav , the RxNorm API and RxMix Acknowledgments collaborators. 2013.

[Bre01] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001. URL: https://doi.org/10.1023/A:1010933404324, doi:10.1023/A:1010933404324.

[cit12] RapidMiner. Online, April 2012. URL: http://rapid-i.com/content/view/181/190/.

[CRI08] KDD , SEMMA AND CRISP-DM : A PARALLEL OVERVIEW Ana Azevedo and M . F . Santos. jan 2008.

[CXP+14] Mei-Chun Cai, Quan Xu, Yan-Jing Pan, Wen Pan, Nan Ji, Yin-Bo Li, Hai-Jing Jin, Ke Liu e Zhi-Liang Ji. ADReCS: an ontology database for aiding standardization and hierarchical classification of adverse drug reaction terms. *Nucleic Acids Research*, 43(D1):D907–D913, oct 2014. URL: https://doi.org/10.1093/nar/gku1066, doi:10.1093/nar/gku1066.

[DDmE+08] Kirill Degtyarenko, Paula De matos, Marcus Ennis, Janna Hastings, Martin Zbinden, Alan Mcnaught, Rafael Alcántara, Michael Darsow, Mickaël Guedj e Michael Ashburner. ChEBI: A database and ontology for chemical entities of biological interest. *Nucleic Acids Research*, 36(SUPPL. 1):344–350, 2008. doi:10.1093/nar/gkm791.

[DKSL13] Lian Duan, Mohammad Khoshneshin, W. Nick Street e Mei Liu. Adverse drug effect detection. *IEEE Journal of Biomedical and Health Informatics*, 17(2):305–311, 2013. doi:10.1109/TITB.2012.2227272.

[GSZ07] Jan E. Gewehr, Martin Szugat e Ralf Zimmer. BioWeka - Extending the Weka framework for bioinformatics. *Bioinformatics*, 23(5):651–653, 2007. doi:10.1093/bioinformatics/btl671.

[Has95] Mohamad H. Hassoun. *Fundamentals of Artificial Neural Networks*. MIT Press, Cambridge, MA, USA, 1st edition, 1995.

[HKP12] Jiawei Han, Micheline Kamber e Jian Pei. *Data Mining: Concepts and Techniques*. 2012. arXiv:arXiv:1011.1669v3, doi:10.1016/B978-0-12-381479-1.00001-0.

[IBM11] IBM. IBM SPSS Modeler CRISP-DM Guide. *IBM Corp*, page 53, 2011.

[Int13] International Conference on harmonisation of technical requirements for registration of pharmaceuticals for human use. Understanding MedDRA:The Medical Dictionary for Regulatory Activities. *International Conference on harmonisation*, 2013.

[KHXM+16] Taha A. Kass-Hout, Zhiheng Xu, Matthew Mohebbi, Hans Nelsen, Adam Baker, Jonathan Levine, Elaine Johanson e Roselie A. Bright. OpenFDA: An innovative platform providing access to a wealth of FDA's publicly available data. *Journal of the American Medical Informatics Association*, 23(3):596–600, 2016. doi:10.1093/jamia/ocv153.

[KLJB15] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen e Peer Bork. The SIDER database of drugs and side effects. *Nucleic Acids Research*, 44(D1):D1075–D1079, oct 2015. URL: https://doi.org/10.1093/nar/gkv1075, doi:10.1093/nar/gkv1075.

[KSK+16] Minoru Kanehisa, Yoko Sato, Masayuki Kawashima, Miho Furumichi e Mao Tanabe. KEGG as a reference resource for gene and protein annotation. 44(October 2015):457–462, 2016. doi:10.1093/nar/gkv1070.

[KTB+16] Sunghwan Kim, Paul A. Thiessen, Evan E. Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A. Shoemaker, Jiyao Wang, Bo Yu, Jian Zhang e Stephen H. Bryant. PubChem substance and compound databases. *Nucleic Acids Research*, 44(D1):D1202–D1213, 2016. doi:10.1093/nar/gkv951.

[MBL+04] Karin Martin, Bernard Bégaud, Philippe Latry, Ghada Miremont-Salamé, Annie Fourrier e Nicholas Moore. Differences between clinical trials and post-marketing use. *British Journal of Clinical Pharmacology*, 57(1):86–92, 2004. doi:10.1046/j.1365-2125.2003.01953.x.

[MC14] Kwankaew Meesuptaweekoon e Paveena Chaovalitwongse. Dynamic vehicle routing problem with multiple depots. 2014.

[ML11] Sérgio Moro e Raul M S Laureano. Using Data Mining for Bank Direct Marketing: An application of the CRISP-DM methodology. *European Simulation and Modelling Conference*, (Figure 1):117–121, 2011.

[MS09] Author Manuscript e Tract Structures. NIH Public Access. 6(3):247–253, 2009. arXiv:NIHMS150003, doi:10.1111/j.1743-6109.2008.01122.x.Endothelial.

[OBJ+11] Noel M OBoyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch e Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1):33, 2011. URL: https://doi.org/10.1186/1758-2946-3-33, doi:10.1186/1758-2946-3-33.

REFERÊNCIAS

[PCCC15] Diogo Pinto, Pedro Costa, Rui Camacho e Vítor Santos Costa. Predicting drugs adverse side-effects using a recommender-system. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9356:201–208, 2015. doi:10.1007/978-3-319-24282-8_17.

[Pet09] L. E. Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009. revision #136646. doi:10.4249/scholarpedia.1883.

[Qui93] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[RSt15] RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA, 2015. URL: http://www.rstudio.com/.

[SC08] Ingo Steinwart e Andreas Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[Sch13] Robert E. Schapire. Explaining adaboost. *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*, pages 37–52, 2013. doi:10.1007/978-3-642-41136-6_5.

[SSvM+15] Damian Szklarczyk, Alberto Santos, Christian von Mering, Lars Juhl Jensen, Peer Bork e Michael Kuhn. STITCH 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic Acids Research*, 44(D1):D380–D384, nov 2015. URL: https://doi.org/10.1093/nar/gkv1277, doi:10.1093/nar/gkv1277.

[WFG+17] David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, Nazanin Assempour, Ithayavani Iynkkaran, Yifeng Liu, Adam Maciejewski, Nicola Gale, Alex Wilson, Lucy Chin, Ryan Cummings, Diana Le, Allison Pon, Craig Knox e Michael Wilson. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46(D1):D1074–D1082, nov 2017. URL: https://doi.org/10.1093/nar/gkx1037, doi:10.1093/nar/gkx1037.

[WFHP16] Ian H. Witten, Eibe Frank, Mark A. Hall e Christopher J. Pal. *Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 4th edition, 2016.

[Wor11] World Health Organization (WHO). ICD-10 Transition. *Family practice management*, 18:39, 2011. URL: http://www.ncbi.nlm.nih.gov/pubmed/22184833, doi:10.1159/000371811.

[Yap10] Chun Wei Yap. PaDEL-descriptor: An open source software to calculate molecular descriptors and fingerprints. *Journal of Computational Chemistry*, 32(7):1466–1474, dec 2010. URL: https://doi.org/10.1002/jcc.21707, doi:10.1002/jcc.21707.

[ZQC15] Hui Zeng, Chengxiang Qiu e Qinghua Cui. Drug-Path: A database for drug-induced pathways. *Database*, 2015:1–4, 2015. doi:10.1093/database/bav061.

REFERÊNCIAS

Anexo A

Apêndice

A.1 parserjsonbd.py

```
1  import json
2  import mysql.connector
3  import re
4  import sys
5
6
7  def main():
8      cnx = mysql.connector.connect(user='root', password='password',
9                                    host='127.0.0.1',
10                                   database='drugs',
11                                   charset='utf8')
12
13     cursor = cnx.cursor()
14
15     with open(sys.argv[1]) as f:
16         f = re.sub(r'\,(?!\s*?[\{\[\"\'\w])', "", f.read())
17         data = json.loads(f)
18         i = 0
19         try:
20             while i < len(data["patient"]["drug"]):
21                 substance_name = data["patient"]["drug"][i]["medicinalproduct"]
22                 add_substance = ("INSERT INTO substances "
23                                  "(name, smiles) "
24                                  "VALUES (%s, %s) ON DUPLICATE KEY UPDATE name=name
                                        ;",
25                                  (substance_name.encode('utf-8'), ''))
26                 cursor.execute(*add_substance)
27                 print ('Substance added')
28                 x = 0
29                 if len(data["patient"]["drug"][i]["openfda"]["rxcui"]) < 5:
30                     while x < len(data["patient"]["drug"][i]["openfda"]["rxcui"]):
```

```
31                          substance_rxcui = data["patient"]["drug"][i]["openfda"]["
                                rxcui"][x]
32                          add_rxcui = ("INSERT INTO subrxcui "
33                                       "(rxcui, substance) "
34                                       "VALUES (%s, %s) ON DUPLICATE KEY UPDATE rxcui
                                             =rxcui;",
35                                       (substance_rxcui.encode('utf-8'),
                                             substance_name.encode('utf-8')))
36                          cursor.execute(*add_rxcui)
37                          print ('Rxcui added')
38                          x = x + 1
39                  else:
40                      while x < 5:
41                          substance_rxcui = data["patient"]["drug"][i]["openfda"]["
                                rxcui"][x]
42                          add_rxcui = ("INSERT INTO subrxcui "
43                                       "(rxcui, substance) "
44                                       "VALUES (%s, %s) ON DUPLICATE KEY UPDATE rxcui
                                             =rxcui;",
45                                       (substance_rxcui.encode('utf-8'),
                                             substance_name.encode('utf-8')))
46                          cursor.execute(*add_rxcui)
47                          print ('Rxcui added')
48                          x = x + 1
49                  i = i + 1
50          except KeyError:
51              print('Erro')
52          except Exception, err:
53              sys.stderr.write('ERROR: %sn' % str(err))
54              print '\n'
55      cnx.commit()
56      cursor.close()
57      cnx.close()
```